Vanskelig på en intuitiv måte?

- En semiotisk analyse av Dreamweavers grafiske brukergrensesnitt

Hege Johanne Tafjord

Masteroppgave i Medievitenskap

Institutt for Medier og Kommunikasjon

UNIVERSITETET I OSLO

13.05.2009

SAMMENDRAG

Er det grafiske brukergrensesnittet i Dreamweaver intuitivt? I oppgaven argumenterer jeg for at intuitivitetsidealet, slik Apple uttrykker det gjennom sine retningslinjer, ikke lar seg realisere på grunn av det innholdet Dreamweavers grensesnitt skal kommunisere. Jeg stiller som novise i møtet med Dreamweaver (et program for å lage og vedlikeholde websites), og prøver å finne ut hva som legger grunnlaget for forståelse/ikke forståelse, sett fra et semiotisk perspektiv. Jeg kommer frem til at forståelse bygges ved gjenbruk av Apples generelle standard elementer (som vinduer, menyer, ikoner osv), samt bruk av et allment vokabular og gjenkjennelige ikoner jeg kan knytte til et innhold. Men den allmenne kompetansen strekker ikke til for å forstå Dreamweavers spesielle funksjonalitet, og dette henger sammen med noen grunnleggende egenskaper ved innholdet. En viktig årsak er at funksjonaliteten i Dreamweaver ikke enkelt lar seg modellere etter en kjent analogi eller metafor, noe som fører til en manglende overhengende konseptuell forståelse av domenet applikasjonen representerer. En annen årsak er at det mangler et allment vokabular som på liten plass presist kan forklare funksjonaliteten. Et siste moment er at ikonene i Dreamweaver generelt er vanskelige å forstå fordi funksjonaliteten som representeres ikke har egenskaper som lar seg vises ikonisk. I sum betyr dette at innhold som er delt mellom applikasjoner og som bygger på felles konvensjoner er lett å forstå, mens spesialisert, abstrakt funksjonalitet knyttes til tegn som aktivt må læres.

ABSTRACT

Is the Graphical User Interface in Dreamweaver intuitive? In the thesis I argue that the ideal of the intuitive user interface, set forth by Apple, is impossible to implement given the content Dreamweaver´s interface has to communicate (how to make a website with this application). In the role of a novice approaching the application, I put on my semiotic glasses in order to figure out what makes me understand or not understand the interface. I conclude that the recycling of Apple´s general interface elements (windows, icons and menus) and the use of a commonly known vocabulary and recognizable icons with a known content, ensures some degree of understanding. But a common knowledge does not suffice in order to understand the specific functionality of Dreamweaver, and this is due to some basic features of Dreamweaver´s functionality. First of all, the content is not easily modeled by some analogy or metaphor from the”real world”. Therefore I am lacking an overarching conceptual understanding of the domain the application represents. Second, the specialized functionality has a specialized vocabulary I do not understand. Third, much of the content is too abstract and complex to be visualized by icons. That is, there are no suitable properties to project. In sum this means that shared content among applications is based on easily understood conventions, while specialized content without easily recognizable analogies to common knowledge has to be actively learned.

FORORD

Denne oppgaven ble født i et frustrert øyeblikk da jeg og datamaskinen min slett ikke forsto hverandre. Det slo meg at det enten var jeg som var idiot, eller maskinen feilet stort i å uttrykke seg. I det jeg nektet å akseptere det første var prosjektet en realitet.

Det er mange som har bidratt til dette tålmodighetsprosjektet på ulikt vis. Tusen takk til min veileder Dagny Stuedahl (VH05,VH06,VH08,V09) for evnen til å stille de viktige og riktige spørsmålene. Takk til Hanne som alltid sier ja. Takk til Goa og Kari som vet å sette farge på dagen. Takk til middagscrewet Lian, Ho, Krzys og Booken som aldri lot meg glemme at jeg burde vært ferdig for lenge siden. Takk til mamma og pappa for uendelig tålmodighet. Og sist, men ikke minst, hjertelig takk til Mona for et skarpt språkblikk og underholdende lunsjer.

1. INNLEDNING

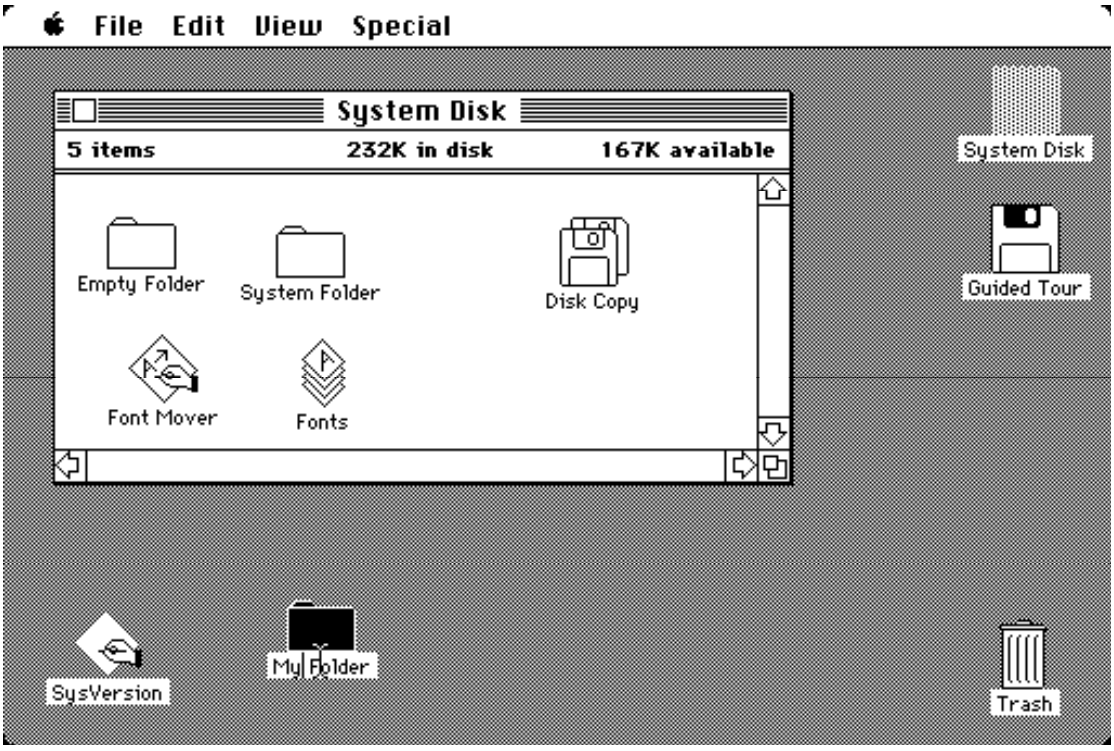
1.1 Kommersialiseringen av det grafiske brukergrensesnittet

I 1984 lanserte Apple sin første Macintosh med brask og bram gjennom en storstilt reklamekampanje. Den påkostede reklamefilmen ”1984”, regissert av Ridley Scott, ble vist for første gang på TV under Super Bowl XVIII i USA, og ble startskuddet for den første Macintoshens kommersielle suksess. En av nyvinningene som ble slått stort opp var det grafiske brukergrensesnittet og den direkte interaksjonsformen ved hjelp av en mus. Riktignok var Lisa introdusert av Apple året før som den første kommersielt tilgjengelige datamaskinen med disse egenskapene, men Macintoshen var billigere og tok steget videre med mer avansert grafikk og det som ble ansett som større brukervennlighet. Lanseringen av den første Macintoshen har gått over i historien som det grafiske brukergrensesnittets kommersielle gjennombrudd (Campbell-Kelly 2004, Kluge 2005).

Tittelen til reklamefilmen ”1984” henspiller på George Orwells berømte roman, og budskapet i reklamekampanjen kan oppsummeres med slagordet ”And you will see why 1984 wont be like ´1984´”. I filmen gjør heltinnen i farger opprør mot den grå totalitære staten ved å kaste ei slegge mot en gigantisk Big Brother liknende skjerm med et grått bilde av regimets leder. En vanlig tolkning er at dette opprøret i stor grad henviste til datidens gigant IBM (the big blue), og at den nye Macintoshen fra Apple skulle sikre fremtiden mot Orwells dystre fremtidsskildring av nettopp 1984. Det demokratiske argumentet henspilte både på det grafiske brukergrensesnittet konstruert rundt WIMP-paradigmet: Windows, Icon, Menu og Pointing device (alternativt: Windows, Icons Mouse, og Pull-down menus), og det faktum at Apples inntreden som konkurrent ville ruske opp i monopoltendensene og utfordre eksisterende maktrelasjoner.

Hva var det så med det nye brukergrensesnittet som kunne forsvare bruken av begrepet demokrati? Først og fremst var det en idé om allmenn tilgjengelighet. Det skulle ikke kreves en dypere forståelse av datamaskinens virkemåte eller omfattende opplæring for å kunne interagere med systemet. Det nye paradigmet, konstruert rundt de grunnleggende WIMP-elementene og logisk organisert rundt kjente metaforer,

skulle være intuitivt og derfor for alle (Preece et al, 2002, s. 60). Macen skulle være en folkelig maskin med lav brukerterskel. En grunnleggende ide var at brukeren ikke lenger skulle trenge å huske vanskelige kommandoer med mange parametre, men heller kunne interagere med datamaskinen ved hjelp av gjenkjennelse; ”point and click” istedenfor ”remember and type”. Den grafiske uttrykksformen ble altså supplert med en mer direkte manipulasjonsform, som f.eks. når en trekker et dokument over i søppelkassen ved hjelp av musen. Det grafiske brukergrensesnittet var det første skrittet på veien mot datamaskinen som multimedia slik vi kjenner den idag (Jørgensen, 2004, s.19).



Bilde 1, Macintosh desktop fra 1984 (Wikipedia, 2009)

I tillegg til det grafiske brukergrensesnittet var det andre viktige faktorer som virket inn på utbredelsestempoet til den personlige datamaskinen. Mikroprosessoren senket prisene på hardware slik at enkeltindivider kunne kjøpe seg en egen datamaskin. Softwareindustrien fikk økt innflytelse, og synet på programvare forandret seg fra oppfatningen om at brukeren selv skulle programmere sin datamaskin, til oppfatningen om at brukeren ønsket ferdigsydde pakkeløsninger med programvare. I sum bidro alle disse tendensene til populariseringen av datamaskinen som en

personlig maskin alle skulle kunne bruke. I dag bruker vi datamaskinen til et mangfold av ulike formål, en tendens som setter store krav til det grafiske brukergrensesnittet som oversetter og forståelsesfilter til datamaskinens virkemåte. Med et slikt mangfold av bruksmuligheter er det interessant å se på hvilket interaksjons- og designideal som gjør seg gjeldene.

1.2 Problemstilling

Denne oppgaven handler om de ulike typer betydningselementer i ulike sammenføyninger som visuelt viser seg på dataskjermen i applikasjonen Dreamweaver, et program utviklet av Macromedia (i dag Adobe) for utvikling, redigering og vedlikehold av websites og webapplikasjoner. Mitt hovedanliggende er å finne ut hvordan det dominerende WIMP-paradigmet, slik det beskrives gjennom Apples retningslinjer for design, fungerer som rammeverk for å uttrykke Dreamweavers funksjonalitet og bruksmuligheter på den måten Apples egne idealer foreskriver. Det vil si at hvis Dreamweavers grensesnitt bygger på Apples generelle retningslinjer, bør designerne av grensesnittet søke å oppfylle de uttalte idealene intuitivitet, vennlighet, eleganse og kraftfullhet gjennom de retningslinjene Apple stiller til rådighet (Apple, 2008, s. 19). For å avgrense omfanget har jeg konsentrert meg om ett ideal, nemlig intuitivitetsidealet. Dette idealet har jeg, ved hjelp av semiotisk teori, valgt å knytte til forståelse. Med denne modifikasjonen kan hovedproblemstillingen oppsummeres slik:

- Hvordan fungerer Apples retningslinjer for design som rammeverk for å kommunisere Dreamweavers funksjonalitet og bruksmuligheter på en forståelig måte?

Deler vi opp problemstillingen ser vi at det dreier seg om forholdet mellom en uttrykksform der visse føringer er lagt i forhold til noen gitte retningslinjer, og et spesifikt innhold som skal kommuniseres gjennom denne uttrykksformen på en forståelig måte. Analysen vil derfor ta sikte på å beskrive Dreamweavers uttrykksformer med utgangspunkt i Apples designprinsipper for å besvare spørsmålet om Dreamweaver gjør seg nytte av de prinsippene som bygger opp under intuitivitetsidealet. Med utgangspunkt i en slik beskrivelse vil fokuset dreie seg mot å gi en karakteristikk av forholdet mellom Dreamweavers uttrykksformer og det

innholdet som skal kommuniseres slik det kommer frem i Dreamweavers brukerguide (Adobe, 2007). Denne karakteristikken bygger på Ecos semiotiske teori (1979), og hovedmålet er å kunne si noe om hvorfor jeg forstår noen tegn, men ikke andre i grensesnittet til Dreamweaver. I denne analysedelen ønsker jeg altså å relatere min personlige forståelse av Dreamweavers funksjonalitet og bruksmuligheter til karakteristiske trekk ved forholdet mellom uttrykk (Dreamweavers grensesnitt) og innhold (faktiske funksjonalitet). De to forskningsspørsmålene kan oppsummeres slik:

1) Hvordan gjør Dreamweaver seg nytte av Apples designprinsipper for å nærme seg intuitivitetsidealet?

2) Hva sier forholdet mellom Dreamweavers uttrykksform og egenskaper ved det funksjonelle innholdet om grunnlaget for forståelse?

Tatt i betraktning at Dreamweaver rommer et stort omfang av funksjonalitet i tilknytning til et stort kunnskapsområde, nemlig webdesign, vil jeg begrense omfanget av oppgaven ved å velge ut de delene av det grafiske brukergrensesnittet som egner seg til å illustrere vesentlige poenger i forhold til dragningen mellom domenespesifikke uttrykksbehov og det eksisterende strukturerende WIMP-paradigmet. Dette utvalget vil bli nøyere spesifisert i metodekapittelet (kap. 5.4).

Før jeg kan gi meg i kast med analysen er det nødvendig å definere hvilke betingelser denne bygger på i form av grunnleggende konsepter og valgt perspektiv. Med utgangspunkt i at valget av uttrykksformer og språklige virkemidler delvis avhenger av mediets egenart, altså egenskaper ved datamaskinen i seg selv og dens virkemåte, vil jeg i bakgrunnskapittelet (kap.2) ta relevante egenskaper ved datamaskinen nærmere i øyensyn og definere de grunnleggende konseptene denne forstås ut fra. Samtidig er det viktig å se at dagens brukergrensesnitt bygger på konvensjoner med lange historiske røtter. I kapittelet som beskriver det empiriske grunnlagsmaterialet og antakelser angående studieobjektet (kap. 3) vil jeg derfor grovt skissere WIMP-paradigmets historie samt gi en innføring i Apples designprinsipper for applikasjoner på Mac OS X plattformen. Dette kapittelet vil også romme en beskrivelse av applikasjonen Dreamweaver og argumentere for valget av denne. Teorikapittelet (kap.4) tar for seg det semiotiske rammeverket som brukes i analysen av hvordan

Dreamweavers grafiske brukergrensesnitt implementerer ”intuitivitet”. Metodekapittelet (kap. 5) utleder fra den semiotiske teorien visse metodiske implikasjoner og diskuterer disse i forhold til analysestrategi. I tillegg beskrives gangen i analysen i mer detalj. Analysekapittelet (kap. 6) struktureres etter den semiotiske teoriens beskrivelse av signifikasjonssystemer, og etablerer to fokusområder. Det ene dreier seg om syntaktiske og strukturelle kjennetegn ved det grafiske brukergrensesnittet til Dreamweaver, mens det andre dreier seg om å karakterisere korrelasjonen mellom uttrykk og innhold, samt graden av forståelse slik jeg opplever det. Analysen av strukturelle kjennetegn og korrelasjonen mellom uttrykk og innhold vil kunne si noe om hvilke designprinsipper som brukes i Dreamweavers grafiske brukergrensesnitt og grunnlaget for forståelse. Slik sett blir både Apples retningslinjer og Dreamweavers grafiske brukergrensesnitt analysert med et semiotisk blikk.

Avslutningsvis vil konklusjonen (kap. 7) ta for seg det overhengende spørsmålet angående dragningen mellom tilpasning til paradigmet (konvensjoner), domenespesifikke uttrykksbehov, og forholdet til intuitivitetsidealet.

2. BAKGRUNN

2.1 Grunnleggende konsepter

I det følgende vil jeg presentere noen sentrale konsepter og begreper som danner bakgrunn for resten av oppgaven. I tillegg vil jeg beskrive egenskaper ved datamaskinen som legger føringer på uttrykksformene i grensesnittet.

Jeg har allerede benyttet meg av begrepet paradigme i forbindelse med omtalen av WIMP. Begrepet WIMP-paradigme er vanlig å bruke i litteratur angående grafiske brukergrensesnitt og er ment å betegne en type filosofi eller måte og tenke på angående interaksjonsdesign (Raskin 2000, Preece et. al 2002).[1] Store norske leksikon skriver om paradigmebegrepet: ”En problemløsning som blir akseptert som forbilledlig for løsninger av lignende problemer innen samme vitenskap, og som derved skaper en vitenskapelig tradisjon” (Store norske, 2009). I forhold til uttrykksformer er det altså en helhetlig ide om hvordan interaksjon mellom maskin og menneske bør foregå og hvilke teknikker som bør benyttes for å realisere idealet.

Domenebegrepet er ment å betegne det kunnskapsområdet en applikasjon representerer og brukes i forhold til. Dreamweaver representerer domenet webdesign, et komplekst kunnskapsområde i stadig utvikling. Domenebegrepet blir nyansert i teorikapittelet (kap.4).

2.2 Datamaskinens virkemåte og egenskaper

Datamaskinens enestående fleksibilitet når det kommer til mulige bruksområder er basert på evnen til symbolsk manipulasjon. For å forstå denne egenskapen kan det være nyttig å se på datamaskinens organisering som en stakk av abstraksjonslag der sifferne 0 og 1 representerer de grunnleggende elektriske signalene og videre opp til høyeste abstraksjonsnivå, som ofte er det grafiske brukergrensesnittet. Datamaskinen benytter seg av kodestandarder for å prosessere den digitale bit-strømmen slik at det

[1] Et søk i ”The ACM Digital Library” under ACM-portalen (publisert av Association for Computing Machinery) på ”WIMP paradigm” ga 183 treff, noe som indikerer begrepets relevans.
http://portal.acm.org/

aldri er mer enn én tolkning med en gitt konverteringsregel. Denne grunnleggende egenskapen, å kunne oversette en serie av to diskrete verdier (0 og 1) til abstrakte symboler og omvendt, gjør datamaskinen enestående fleksibel og tilpasningsdyktig.

Det er grunnleggende sett stor forskjell mellom symbolene i grensesnittet og symbolene i de underliggende lagene. Denne forskjellen bunner i vesentlig grad i hvordan den menneskelige hjerne fortolker symboler kontra maskinell symbolprosessering. I maskinen er forløpet av hendelser forutsett gjennom programkoden og prosessorens kode-eksekvering følger slavisk entydige instruksjoner.[2] Overgangen mellom maskinell tegnprosessering og menneskelig fortolkning kan sees som en slags terskel som markerer en avgrensning av semiotikkens interesseområde. Den menneskelige fortolkningen baserer seg også på et kodesystem, men dette er i evig forandring og ulike fortolkninger kan springe ut av samme kodegrunnlag avhengig av kontekst og andre omstendigheter.

2.3 Tegn og tegnfunksjoner

Denne oppgaven baserer seg på Umberto Ecos semiotiske teori slik den er formulert i boka ”A theory of Semiotics” fra 1979. Språklig sett henger Umberto Ecos bruk av begrepene tegn og tegnfunksjon sammen med de to hoved-delene i hans semiotiske teori, nemlig de som omhandler tegnproduksjon og kodeteori. Begrepet tegnfunksjon benyttes i beskrivelsen av kodeteorien for å nyansere og presisere tegnbegrepet i et kodeperspektiv, mens tegnbegrepet hører til den delen av den semiotiske teorien som fokuserer på prosess i form av tegnproduksjon. Semiotikk dreier seg om alt som kan forstås som tegn av en menneskelig fortolker, og Umberto Eco definerer et tegn som ”everything which can be taken as significantly substituting for something else”(Eco, 1979, s.7). I forlengelse av tegnets funksjon å ”stå for noe annet” må det understrekes at dette ”noe annet” ikke nødvendigvis trenger å være sant. Vi kan i tråd med Eco derfor beskrive semiotikkens studieobjekt som alt som kan brukes for å lyve. Eco skriver:”the discipline studying everything which can be used in order to lie” (ibid,

[2] Hendelser det ikke er tatt høyde for fører gjerne til at applikasjonen terminerer eller blir gående i evig loop (henger). For en overbevisende argumentasjon angående datamaskinens begrensninger i forhold til menneskelig beslutningsevne se forøvrig Weizenbaum (1976).

s.58, 59). Med denne definisjonen er samtidig nok en terskel i forhold til semiotikkens interesseområde definert, nemlig den mellom vilkår for signifikasjon (vilkår for eksistensen av og bruken av et kodesystem for kommunikasjon) og vilkår for sannhet. Eco beskriver sin semiotiske teori som ”A logic of culture”, noe som innebærer at tegnets innhold først og fremst er kulturelle enheter og at spørsmålet om disse enhetenes forhold til sannhet ikke vedgår semiotikken (ibid, s.3). Det grafiske brukergrensesnittet henvender seg til brukeren av datamaskinen gjennom tegn som må fortolkes, det er derfor godt egnet for semiotisk analyse.

2.4 Det grafiske brukergrensesnittet

I ytterste konsekvens er datamaskinens fleksible egenart fåfengt hvis ingen vet å benytte seg av den. Øverst i abstraksjonshierarkiet finnes derfor grensesnittet brukeren forholder seg til og interagerer med. Bokmålsordboka avgrenser begrepet brukergrensesnitt ved å definere det som: ” programvare som gjør det mulig for en bruker å kommunisere med en datamaskin” (Bokmålsordboka, 2006). For at jeg skal kunne bruke datamaskinen må jeg altså kunne kommunisere mine intensjoner gjennom et brukergrensesnitt på en måte som følger de interaksjonsstandardene grensesnittet legger opp til. Denne kommunikasjonen foregår ved hjelp av et spesifikt språk i grensesnittet, designet for å formidle et spesifikt budskap. I tillegg til programvaredelen vil jeg definere brukergrensesnitt til også å omfatte den rent fysiske delen, oftest tastatur, mus og skjerm. Jeg forholder meg for eksempel i dette øyeblikk til det grafiske brukergrensesnitt i Microsoft Word (programvaredelen) slik det viser seg på min dataskjerm, og bruker et tastatur for å skrive bokstaver og mus for å velge ulike operasjoner systemet tilbyr meg. Slik sett former og regulerer grensesnittet interaksjonen mellom meg og datamaskinen.

Det forutsettes altså at datamaskinen kan kommunisere, at det ligger en intensjon bak tegnene. Grunnleggende for den semiotiske analysen er idéen om at tegnene i brukergrensesnittet er resultat av bevisst menneskelig tegnproduksjon og at brukergrensesnittet derfor visualiserer designerteamets forståelse av datamaskinen,

dataprosessene og domenet applikasjonen gjelder.[3] Men intensjoner kan ha mange uttrykk, og valg av uttrykksform har konsekvenser for *hvordan* kommunikasjonen foregår. Kommunikasjon i forhold til et grafisk brukergrensesnitt sikter til vekselvirkningen mellom fortolkning av tegnene på skjermen og interaksjon ved hjelp av mus eller tastatur (eventuelt andre tilgjengelige midler) med de samme tegnene for å formidle en intensjon. I brukergrensesnittet formidles datamaskinens virkemåte gjennom tegn med mening; interaksjonsforholdet er altså basert på interpretasjon hos bruker, tegnproduksjon (hos bruker) i det intensjoner formidles, samt signalprosessering i datamaskinen.

2.5 Interaksjon

Den menneskelige fortolkningen av tegnene i brukergrensesnittet legger grunnlaget for interaksjon mellom menneske og datamaskin, men også for hvordan brukeren forstår datamaskinen i seg selv. Begrepet ”interaksjon” går ofte igjen i tekster som omhandler datamaskinen, og muligheten for interaksjon blir gjerne betegnet som et vesentlig kjennetegn (Kluge 2005, Manovich 2001, Krippendorff 2006). Det er omdiskutert hva begrepet egentlig betegner, og om det er brukt så mye at det kan bety alt og ingenting (Manovich 2001, Fagerjord 2003). Ofte har det ulikt innhold i ulike fagtradisjoner. I Kunnskapsforlagets blå fremmedordbok er begrepet definert som ”samhandling, vekselvirkning, gjensidig påvirkning” (Kunnskapsf. fr.ordbok, 2004). I denne oppgaven betegner det den kommunikasjonen mellom datamaskin og bruker som legger grunnlaget for aktive valg og fysisk handling. Det vil si at kommunikasjon er en forutsetning for å utløse interaksjon, og at interaksjon først og fremst er knyttet til de *reaktive* tegnene. Kommunikasjon foregår der tegnene på skjermen blir fortolket av brukeren, men interaksjon forekommer først når fortolkningen innebærer en reaksjon i form av en handling hos brukeren.

Interaksjon som et kjennetegn på datamaskinen gjenspeiles også i et språklig skifte fra ”publikum” eller ”konsument” til ”bruker”. Ved hjelp av begrepet ”bruker” hviskes

3 Begrepene ”designer” og ”designerteam” er gjennomgående brukt om de som står bak det grafiske brukergrensesnittet i Dreamweaver, eller den eller de som uttaler seg på vegne av dette teamet. Det må altså ikke forveksles med brukeren eller brukerne av applikasjonen.

skillet mellom produsent og konsument ut. En ”bruker” er i dag begge deler; både konsument og produsent.

I neste avsnitt skal vi se at datamaskinens mangfold av bruksmuligheter gir seg utslag i ulike kommunikasjonsroller avhengig av om grensesnittet skal formidle interaksjonsalternativer eller representere annet type innhold.

2.6 Den dobbelte kommunikasjonsrollen: kontroll versus representasjon

Vi har i dette kapittelet sett at datamaskinens virkemåte gjennom manipulasjon av symboler gjør den enestående fleksibel, og at det er det grafiske brukergrensesnittets oppgave å formidle mangfoldet av bruksmuligheter. Denne enestående fleksibilitet når det kommer til mulige bruksområder fører med seg et mangfold av uttrykksformer, men også visse kjennetegn på tvers av ulikt innhold. Generelt skilles det gjerne mellom datamaskinen som produktivt *verktøy* og uttrykksmessig *media*. Dette impliserer en dobbelt kommunikasjonsrolle avhengig av hva som primært skal kommuniseres, men de to rollene mikses gjerne i varierende grad. Noen tegn står for handlingsalternativ og er kausalt knyttet til en algoritme som kan aktiviseres ved nettopp dette tegnet. Jeg velger å kalle disse tegnene *reaktive*, og de kommuniserer først og fremst et handlingsalternativ knyttet til en funksjon. Andre tegn representerer et budskap i tradisjonell forstand, de representerer ikke noe interaksjonsalternativ. Det visuelle uttrykket på dataskjermen har da funksjon som et endelig produkt, og tegnene referer gjerne til noe utenfor datamaskinen selv. De to rollene kan forstås som kontroll versus representasjon, og pakkes gjerne inn med begrepene verktøy versus media (Manovich, 2001, s. 88). Rammen som settes, det vil si om datamaskinen forstås som verktøy eller medium i interaksjonsøyeblikket, utgjør en viktig del av konteksten rundt interaksjonsforholdet og har konsekvenser for uttrykksformen.

Verktøymetaforen som konseptuell modell etablerer en kontekst som er velegnet for å formidle hvordan datamaskinen kan brukes til å løse ulike oppgaver og dermed gi tegnene på skjermen rollen som kontrollpanel. Brukergrensesnittet fungerer da funksjonelt som et virtuelt redskap/instrument brukeren kan benytte seg av for å utføre ulike operasjoner. Tegnene er i denne konteksten hovedsakelig reaktive; de utløser en reaksjon ved bruk.

Samtidig indikerer verktøymetaforen en idé om at det grafiske brukergrensesnittet ikke skal tiltrekke seg oppmerksomhet i seg selv. Som tradisjonelle verktøy skal det være mest mulig intuitivt, det vil si at brukeren skal interagere med tegnene på skjermen uten å tenke på det. Derfor må tegnene i størst mulig grad ”gi seg selv”, slik vi opplever at for eksempel formen på hammeren gjør det. Andersen skriver om det optimale grensesnittet i verktøymodus: ”Given the concepts of operations and actions, a good tool can now be characterized as a tool that lets the user direct her attention towards the object of work and makes the tool itself disappear. Thus, the tool should be handled by unconscious operations, while the work object can be manipulated by conscious actions” (Andersen, 1990, s.25). I avsnittet angående dagens designparadigme skal vi se at tradisjonelle verktøys evne til å være selvforklarende utgjør et ideal i utarbeidelsen av grafiske brukergrensesnitt i dag. Vi skal samtidig se at Ecos semiotiske teori kan brukes til å avskrive drømmen om den intuitive forståelse. Som følge av hans definisjon av tegn som stående for noe annet, kan vi si at hammeren står for aktiviteten å hamre (m.m.). Den er derfor et tegn og dermed tuftet på en konvensjonell regel som knytter uttrykk til innhold. At noe føles ”intuitivt” er kun fordi vi kan det godt (Eco, 1979).

Men datamaskinens verktøy skiller seg på viktige områder fra de tradisjonelle verktøyene. Eksekvering av algoritmer krever ikke menneskelig fysisk styrke, og brukeren har som regel ikke oversikt over detaljene rundt utførelsen av oppgaven. Det er også umulig å si noe om hvilke bruksområder datamaskinen har ut fra dens form; funksjonaliteten må kommuniseres ved hjelp av tegn. Interaksjonsforholdet er derfor basert på semantikk i motsetning til tradisjonelle verktøy som gjerne krever fysisk styrke og som har en form som betegner en bestemt funksjonalitet. Brukeren av datamaskinens verktøy må forstå tegnene gjennom interpretasjon og selv interagere med brukergrensesnittet for å uttrykke egne intensjoner. Ulikt annen type verktøy er programvare altså styrt gjennom symbolsk manipulasjon og interpretasjon. Til tross for alle disse forskjellene er verktøymetaforen som konseptuell modell særdeles nyttig og vanlig for å forklare hva en applikasjon er, og ramme inn interaksjonsforholdet deretter.

Den dobbelte kommunikasjonsrollen er et vesentlig karaktertrekk ved datamaskinens grafiske brukergrensesnitt, og en ser ulik vektlegging av de to rollene etter hvilke innhold designen uttrykker. Det mest åpenbare er å betegne Dreamweaver som et verktøy, og denne klassifiseringen har, som vi skal se senere i teorikapittelet, konsekvenser for uttrykksformen. Det optimale er å ha kun reaktive tegn som i seg selv både kommuniserer funksjonaliteten og er utløser til funksjonen. Dette fordi tegnene skal brukes om igjen og om igjen, og forståelsen knyttet til enkeltelementer og deres relasjon til andre elementer trenger ofte kun og etableres en gang. Det betyr ikke at forståelsen stivner, men heller utbygges gradvis ved at nye oppdagelser tilpasses gamle. Tatt i betraktning all tilleggsdokumentasjon og opplæringsmateriell som finnes til ulike applikasjoner, er det klart at pedagogiske hensyn og det pragmatiske bruksperspektivet ikke alltid lar seg integrere på optimalt vis. Økt kompleksitet kan føre til vanskeligere tilgjengelighet og setter store krav til designen i brukergrensesnittet. Donald Norman (1988) beskriver dilemmaet angående økt funksjonalitet og vanskeligere tilgjengelighet som et teknologiens paradoks. I følge han består dette paradokset i at ny teknologi tilbyr stadig mer funksjonalitet i form av måter å hjelpe mennesker på i det daglige virke. Men samtidig som nye løsninger blir utviklet blir det stadig vanskeligere å forstå og bruke de nye hjelpemidlene. Den økte kompleksiteten fører til frustrasjon og sinne hos brukerne, altså det motsatte av intensjonen (Norman, 1988, s.29).

Som vi skal komme tilbake til i beskrivelsen av Apples designprinsipper i kapittel 3 er disse nettopp motivert av pedagogiske hensyn i form av å tilby et integrert arbeidsmiljø med gjenkjennelig design på tvers av ulike applikasjoner. Når det kommer til den faktiske designen er det en utfordring å integrere læring og bruk i det grafiske brukergrensesnittet. Dette blir grundigere behandlet i analysekapittelet (kap. 6) der uttrykksformer i Dreamweaver blir vurdert ut fra hvordan de realiserer intuitivitetsidealet.

2.7 Alternative interaksjonsformer

Det er slett ikke opplagt at det er det grafiske brukergrensesnittet på dataskjermen slik vi kjenner det i dag som er beste løsning for interaksjon mellom menneske og datamaskin. Det kan være mange ulike måter å bruke datamaskinen på, og det utvikles stadig nye interaksjonsformer. Nintendos siste spillkonsoll Wii baserer seg for eksempel på interaksjon ved hjelp av spillerens fysiske gester og bevegelser i tillegg til de tradisjonelle knappene, mens mange mobiltelefoner baserer interaksjonen på ”touch-screens”. Men mest vanlig er fortsatt det grafiske brukergrensesnittet slik det presenteres på skjermen til en vanlig personlig datamaskin, og hvor interaksjonen foregår ved hjelp av datamaskinens tastatur og mus.

3. EMPIRISK BESKRIVELSE

I dette kapittelet vil jeg introdusere det empiriske materialet jeg senere skal analysere, nemlig Dreamweaver og de generelle designprinsippene utarbeidet av Apple. De generelle designprinsippene konseptualiseres som empirisk materiale fordi de får en grundig "semiotisk behandling" i analysekapittelet.

3.1 Dreamweaver

Dreamweaver er en applikasjon for utvikling, redigering og vedlikehold av websites og webapplikasjoner. Det grafiske brukergrensesnittet er basert på WIMP-paradigmet, og første versjon av Dreamweaver (Dreamweaver 1.0) ble utgitt av Macromedia i 1997. Siden den gang har en rekke nye versjoner kommet på markedet, med Dreamweaver CS4 som siste tilskudd i 2008. I 2005 ble Macromedia kjøpt opp av Adobe Systems, og dette har ført til fokus på en sterkere integrering mellom tidligere Macromedia produkter og Adobe produkter (Adobe to acquire Macromedia, 2009).

3.1.1 Hvorfor Dreamweaver?

Valget av analyseobjekt er først og fremst basert på ideen om at en applikasjon til bruk i grafisk arbeid vil ligge i front når det gjelder egen design. I tillegg er valget gjort på grunnlag av at Dreamweaver representerer et domene som er i konstant utvikling, særlig med tanke på nye standarder innen webteknologi. Websites får stadig ny funksjonalitet og utseende, og designere og utviklere krever verktøy som kan støtte innovative ideer. I omtalen av Dreamweaver på Adobe sine hjemmesider skilles det mellom designere og utviklere. Dette skillet reflekterer i stor grad utviklingen innen webteknologi hvor utviklere konsentrerer seg om å utvikle mer eller mindre sofistikert funksjonalitet, ofte i forbindelse med dynamiske websites og webapplikasjoner knyttet til databaser. Designere er i større grad fokusert på den estetiske, visuelle representasjonen av denne funksjonaliteten. Verktøy for utvikling av websites må nødvendigvis ha et mangfold av funksjonalitet, og å kommunisere dette til brukeren er en utfordring for designeren(e) av det grafiske brukergrensesnittet i applikasjonen.

Designeren(e) av brukergrensesnittet i Dreamweaver, og utviklerne av applikasjonen, trenger utførlig kunnskap om domenet applikasjonen representerer. Sett fra et

medievitenskapelig ståsted, hvor mye fokus er rettet mot genre og uttrykksformer i websites, kan det kanskje være nyttig å rette blikket også mot uttrykksformer i underliggende programvare. I den grad nye mediers uttrykksformer forandrer vår sanse-evne og erfaring, kan de også sies å ha innflytelse på den produktive praksisen, og omvendt. Akkurat som det eksisterer et samspill mellom Dreamweaver og websites, kan det på mer abstrakt nivå hevdes at teknologi generelt innlemmer kunnskapen brukt til å lage den, og samtidig medierer produksjonen av ny kunnskap (Thurk, Jessica, Gary Alan Fine, 2003).

Dreamweaver som standard applikasjon (med det mener jeg at den er lagret og eksekveres på den personlige datamaskinen), er interessant fordi den på den ene siden skal støtte innovative idéer, ny funksjonalitet og kreative uttrykksformer, mens den på den andre siden selv må tilpasses det etablerte WIMP-paradigmet. I forhold til ny webteknologi som muliggjør ny funksjonalitet, særlig gjennom nettleseren(e) brukeren har installert, frigjør webapplikasjoner seg fra OS-spesifikke retningslinjer angående brukergrensesnittet. Tjenester som tilbys gjennom webapplikasjoner har et utseende mer avhengig av type nettleser som blir brukt enn operativsystem. Webapplikasjoner står derfor friere til å utforske nye uttrykk enn det Dreamweaver selv gjør. Men samtidig er det klart at regelen om å bruke konvensjoner fortsatt gjelder for å maksimere brukervennligheten.

Dreamweaver er velegnet til å beskrive forholdet mellom behovet for kontinuitet og visualisering av ny funksjonalitet knyttet til stadig ny webteknologi. Med mer avansert funksjonalitet knyttet til webapplikasjoner og websites kan det være at det tradisjonelle konseptuelle og uttrykksmessige skillet mellom applikasjoner og media viskes ut slik at problemstillingen som tas opp kanskje kan fange bredere enn kun Dreamweavers domene.

3.2 WIMP-paradigmet i form av Apples retningslinjer

Akkurat som det innenfor krimsjangeren finnes noen generelle kjennetegn som nettopp skiller den fra andre sjangre, finnes det visse felles kjennetegn ved brukergrensesnitt på Mac OS plattformen. Det er disse som er beskrevet i Apples generelle retningslinjer for design. I det følgende vil jeg kort beskrive de enkelte

designprinsippene, mens retningslinjene angående utseende og oppførsel til de enkelte elementene vil integreres i analysen (kap. 6).

Før jeg tar for meg WIMP-paradigmet i form av Apples retningslinjer for design, vil jeg kjapt gi noen stikkord angående WIMP-paradigmets historiske røtter. Sett fra et menneske-maskin interaksjon perspektiv var perioden før 1963 mest interessant på grunn av de innflytelsesrike visjonene som ble fremsatt angående datamaskinens mulige bruksområder. De mest fremtredende visjonærene på denne tiden, Vannevar Bush, J.C.R. Licklider, Douglas Engelbart, Ted Nelson og Ivan Sutherland, har alle hatt innflytelse på dagens dominerende WIMP-baserte grafiske brukergrensesnittet. WIMP står som nevnt for Windows, Icons, Menus og Pointing device, eller alternativt Windows, Icons, Mouse og Pull-down menus. Dette er de grunnleggende elementene i WIMP-paradigmet, som fortsatt råder grunnen for standard applikasjoner utviklet for skrivebordet (desktop), med en enkel bruker sittende foran en datamaskin med skjerm, tastatur og mus (Preece et al, 2002, s.60). De fundamentale særpregene til det grafiske brukergrensesnittet innen WIMP-paradigmet slik vi kjenner det er sterkt influert av gruppen til Douglas Engelbart og deres prototype på det elektroniske kontor NLS (oNLine System), demonstrert på ”the National Computer Conference” i San Francisco i 1968. Sammen med konsepter hentet fra Alan Kay´s Dynabook, det objektorienterte programmeringsspråket Smalltalk og viktige forbedringer innen maskinvare, som høyere skjermoppløsning, rastergrafikk og raskere multiprosessorer, ble grunnlaget lagt for dagens grafiske brukergrensesnitt (Campbell-Kelly 2004, Pew 2008).

Det grafiske brukergrensesnittet i Apples første Macintosh fra 1984 og den påfølgende utgivelsen av ”The Human Interface Guidelines” (Apple, 1987) har hatt stor og vedvarende innflytelse på design av grafiske brukergrensesnitt (Kluge, 2005, s. 51,52). Retningslinjene fra Apple utgjør en slags mal for hvordan applikasjoner som skal kjøres på en Macintosh bør utformes for mest mulig sømløs integrasjon. I det følgende vil jeg ta for meg Apples grunnleggende filosofi og designprinsipper slik de er beskrevet i siste utgivelse av Apple Human Interface Guidelines av 2008 (Apple, 2008). Retningslinjene her er utarbeidet for siste versjon av operativsystemet, altså Mac OS X 10.5.v. Dreamweaver CS3 kom i april 2007, noe som betyr at utviklingen av det grafiske grensesnittet den gang må ha benyttet en tidligere versjon

av Apples retningslinjer. Jeg kjører Dreamweaver på Mac OS X 10.4.11, men dette burde ikke by på problemer da Dreamweaver CS3 er designet for å kunne kjøres på mitt operativsystem også, og da Apple predikerer prinsippet om kontinuitet. En titt i avsnittet ”Revision History” i 2008-utgaven indikerer at det ikke er foretatt avgjørende forandringer, og store omveltninger er heller ikke å forvente i de generelle retningslinjene.

Apples retningslinjer fra 2008 er på hele 386 sider, så jeg må nødvendigvis foreta noen valg basert på relevans i forhold til Dreamweaver og analysen. De originale retningslinjene fra 1987 er kun på 144 sider, så det er tydelig at noe utvikling har det vært. Det fremheves at formålet er å assistere i utviklingen av konsistent visuelt og brukermessig erfaring på tvers av applikasjoner og operativsystem. For å fremme et konsistent arbeidsmiljø, som gavner både utviklere av applikasjoner og Apple, er det ikke så overraskende at retningslinjene er forholdsvis detaljerte. Det pekes på at fordelene ved å følge retningslinjene blant annet er at kontinuitet gjør innlæring og bruk kjappere da applikasjonen ikke ”kommer i veien” for det formålet den understøtter; altså brukergrensesnittet forstått som et ideelt sett mest mulig usynlig verktøy. I tillegg hevdes det at dokumentasjonsarbeidet vil bli enklere da intuitivt grensesnitt og standard oppførsel ikke trenger så mye forklaring. I sum vil etterfølgelse av retningslinjene sikre Macintoshens karaktertrekk; definert som: intuitiv, vennlig, elegant og kraftfull/sterk (powerful) (Apple, 2008).

Apples retningslinjer for design inkluderer både generelle designprinsipper og mer konkrete retningslinjer angående de grafiske elementene. Av praktiske hensyn og for å unngå repetisjon vil et utvalg av de grafiske elementene først bli presentert i analysekapittelet (kap. 6). Dette fordi de er konkretisert gjennom Dreamweavers brukergrensesnitt og er direkte involvert i analysen angående hvordan Dreamweavers funksjonalitet formidles gjennom WIMP-paradigmets karakteristiske uttrykksformer, i tillegg til at de analyseres i forhold til i hvilken grad jeg forstår det innholdet de representerer.

3.2.1 Bruk, brukeren og brukergrensesnittets oppgave

I Apples retningslinjer blir det fokusert på at designfilosofien er bygget rundt den menneskelige faktoren i menneske-maskin forholdet. Begrepet *bruker* er

fremtredende, og dette er en person som ønsker å benytte datamaskinen som et verktøy for å utføre en oppgave mest mulig effektivt. Skrivebordsmiljøet (desktop environment) skal derfor være effektivt, direkte (intuitivt) og tilgjengelig. I kapittelet som gir retningslinjer for designprosessen blir det vektlagt at det er viktig å analysere hvilke oppgaver brukeren ønsker å utføre ved hjelp av applikasjonen som utvikles. Og det er samtidig forutsatt at brukeren allerede vet å utføre oppgaven, og at applikasjonen simpelt hen skal lette prosessen:

> A mental model paints a picture of a task and defines expectations about the components of the task, the organization of those components, and the overall workflow (...). To help you discover the mental models people associate with your product's tasks, look at how they perform similar tasks without a computer (Apple, 2008, s. 26).

De første dataprogrammene effektiviserte allerede eksisterende administrative oppgaver innen store organisasjoner (Weizenbaum, 1976, s.32,33). Dette reflekteres ved bruken av skrivebordsmetaforen og det faktum at WIMP egner seg godt til å abstrahere kontorarbeid, samt dokumenter og funksjonalitet knyttet til dette. Et viktig poeng med å velge Dreamweaver som analyseobjekt er nettopp at det ikke er opplagt hvilke analogier en skal spille på for å forklare applikasjonens domene. Webdesign eksisterer kun i sammenheng med datamaskinen som verktøy og medium. Det er derfor spennende å se hvilke konseptuelle løsninger som er valgt for å forklare applikasjonens arbeidsprosesser og funksjonalitet.

3.3 Generelle designprinsipper

Det overhengende målet med designen er i retningslinjene til Apple beskrevet som et elegant, intuitivt, effektivt og veltilpasset brukergrensesnitt (Apple, 2008, s. 39). I det følgende vil jeg presentere de designprinsippene som understøtter intuitivitetsidealet.

3.3.1 Reflekter brukerens mentale modell[4]

I følge dette designprinsippet har brukeren allerede en mental modell som beskriver oppgaven programvaren er ment å representere. Denne modellen oppstår ved en kombinasjon av erfaringer fra ”den virkelige” verden, erfaring med annen programvare, og med datamaskiner generelt. Brukeren har for eksempel erfaring med å skrive og sende brev fra den ”virkelige” verden, og har derfor en konseptuell modell av denne oppgaven. Brukergrensesnittdesignerens oppgave er dermed å oppdage brukerens mentale modell av oppgaven applikasjonen skal understøtte for så å konstruere brukergrensesnittet i overensstemmelse med denne. En slik modell har gjerne innebygde metaforer som representerer konseptuelle komponenter, som for eksempel brev, postkasser og konvolutter i e-post applikasjoner.

Skrivebordsmetaforen er det mest opplagte eksempelet på en implementert konseptuell metafor. En konseptuell metafor kan baseres på en aktivitet eller et objekt eller begge deler, slik som skrivebordsmetaforen. I dag vil jeg anta få tenker på den opprinnelige betydningen av ordet skrivebord (oftest brukes det engelske ordet ´desktop´ også på norsk) når de forholder seg til det grafiske brukergrensesnittet på dataskjermen. At denne metaforen er rimelig død bekreftes også ved at et oppslag i Kunnskapsforlagets norsk-engelsk ordbok har en egen EDB-spesifisert oversettelse (Kunnskapsf. Engelsk, 2007). En av grunnbetydningene til ordet skrivebord er i følge ordboka altså grafisk grensesnitt. Det er viktig å understreke at metaforer gjør seg nytte av ulike, og ofte begrensede egenskaper ved det opprinnelige innholdet. Metaforen utgjør en modell for forståelse; analogien trenger derfor ikke være fullstendig. Det ville begrense funksjonaliteten i datamaskinen dramatisk. En grundigere beskrivelse av metaforer kommer i designprinsippet angående metaforer (3.3.2) samt i teorikapittelet (kap. 4).

Skrivebordet som konseptuell modell kan også betegnes som en overgangsanalogi. I et historisk perspektiv er det gjerne en viss kontinuitet når det kommer til uttrykksformer i ulike medier. Bruk av overgangsanalogier, det vil si at nye konsepter forklares ved hjelp av allerede etablert kunnskap er derfor et nyttig språklig

[4] Begrepet ”mental modell” er en del av Apples vokabular i retningslinjene. Jeg vil videreføre dette vokabularet i omtalen av retningslinjene.

virkemiddel når nye idéer skal uttrykkes. Teknikken med å bruke analogier til tidligere arbeid i kontorlandskap lettet overgangen til bruk av datamaskinen som hjelpemiddel i kontorarbeidet. Behovet for kontinuitet i uttrykksformene har ført til at de samme analogiene fortsatt ligger som basis i dag; utviklingen av brukergrensesnitt er preget av evolusjon fremfor revolusjon.

3.3.2 Metaforer

Som forklart i forbindelse med konseptuelle metaforer kan brukerens eksisterende kunnskap om verden benyttes ved å bruke metaforer for å uttrykke konsepter og funksjonalitet i applikasjonen, noe som kan gjøre designen mer ”intuitiv”.

Rådet om å bruke metaforer føyer seg inn blant prinsippene som omhandler forholdet mellom innhold og uttrykk. I tillegg er metaforprinsippet relevant som del av den kontekstuelle fortolkningsrammen. Jeg vet jeg kan forvente å finne bruk av metaforer i møte med ukjente grensesnitt som retter seg etter WIMP-paradigmet, og er derfor beredt til å fortolke ukjente og overraskende uttrykksinstanser i overført betydning, samt trekke konvensjonelle slutninger angående de velkjente døde metaforene.

Mange av de klassiske WIMP-elementene kan karakteriseres som døde metaforer, som for eksempel vinduer og menyer.

3.3.3 Eksplisitte og underforståtte handlinger

Hver Mac OS X operasjon involverer manipulasjon av et objekt ved hjelp av en handling.[5] Først ser brukeren det ønskede objektet på skjermen, så velger eller utpeker brukeren dette objektet. Til slutt utfører brukeren en handling, enten ved hjelp av en menykommando eller ved direkte manipulasjon av objektet ved hjelp av mus eller annen ordning. Dette kalles henholdsvis eksplisitte og underforståtte handlinger.

Ved eksplisitte handlinger er resultatet av manipulasjonen av objektet uttrykt spesifikt på forhånd gjennom et tegn, for eksempel ved menyvalg der tilgjengelige kommandoer er listet i form av ord. Navnet på menykommandoen forteller hvilken

[5] Begrepet objekt brukes i Apples manual og referer til brukerproduserte tegn. Jeg vil fortsette å bruke dette begrepet for å opprettholde et språklig skille mellom applikasjonens faste uttrykk og de varierende brukerproduserte objektene.

handling som tilbys og om handlingen er lovlig i den gitte konteksten. Brukeren trenger derfor ikke huske kommandoene som kan utføres på et gitt objekt.

Underforståtte (implied) handlinger tilkjennegir resultatet av en handling gjennom visuelle hint eller kontekst, gjerne i sanntid. En ”drag-and-drop” operasjon er et eksempel på en underforstått handling. For at underforståtte handlinger skal være tydelige må brukeren kjenne igjen objektene som er involvert, manipulasjonen som kan utføres, og konsekvenser av handlingen.

3.3.4 Direkte manipulasjon

Direkte manipulasjon er et eksempel på en underforstått handling som gjør det mulig for brukere å føle at de kontrollerer objektene. I følge dette prinsippet bør et objekt på skjermen fortsette å være synlig mens en bruker utfører en handling på det, og konsekvensen av handlingen bør være umiddelbart synlig. Det vanligste eksempelet er ”drag-and-drop” handlinger, som for eksempel å flytte en fil ved å dra dens ikon fra ett sted til et annet.

3.3.5 Tilbakemelding og kommunikasjon

Tilbakemelding og kommunikasjon dreier seg om å holde brukere informert om hva som skjer ved å gi tilrettelagt tilbakemelding og gjøre mulig kommunikasjon med applikasjonen. Når en bruker initierer en handling skal applikasjonen gi en indikasjon på at den har mottatt og arbeider med brukerens input slik at brukeren er sikker på at kommandoen blir utført. Hvis en kommando ikke kan eksekveres må brukeren få vite hvorfor og hva som kan gjøres isteden. Sparsommelig bruk av animasjon er effektivt for å gjøre klart relasjonen mellom objekter og konsekvenser av handlinger. I tillegg er det viktig at tilbakemeldingene er klare og forståelige. Feilmeldinger skal fortelle akkurat hvilken situasjon som forårsaket feilen og mulige måter å rette den på.

3.3.6 Konsistens

Konsistens i brukergrensesnittet tillater brukere å overføre kunnskap og ferdigheter fra en applikasjon til en annen. Ved å bruke standardelementer i brukergrensesnittet sikres konsistens innen applikasjonen og en drar nytte av konsistens mellom applikasjoner. Brukergrensesnittet bør derfor være konsistens med Mac OS X standard, men også innad i applikasjonen. Det innebærer blant annet konsistent

terminologi på etiketter (label) og funksjoner, at samme ikon har konsistent mening, at konsepter er presentert på konsistent vis, at like kontroller og andre elementer er plassert på samme steder i vinduer og dialoger. I tillegg må det være konsistens med tidligere versjoner av produktet samt folks forventninger og behov.

3.3.7 WYSIWYG (What You See Is What You Get)

I applikasjoner hvor brukere kan formattere data for utskrift, publisering på nett, eller andre formater er det viktig at det ikke er noen avgjørende forskjell mellom hva brukere ser på skjermen og hva de får som avsluttende output. I tillegg må brukere kunne finne all tilgjengelig funksjonalitet i applikasjonen, og kommandoene skal være synlige for gjenkjenning isteden for å måtte huskes. En skal unngå å gi tilgang til funksjonalitet bare i verktøylinjer og kontekstuelle menyer fordi disse kan være skjult. Kommandoer som finnes her skal alltid også være tilgjengelige i menyer.

3.3.8 Tilgivelse

Brukere skal oppfordres til å utforske applikasjonen ved å bygge inn tilgivelse; det betyr å gjøre de fleste operasjoner enkle å reversere. Brukere må kunne føle at de kan bruke applikasjonen uten å ødelegge det eller sette data i fare. Det bør derfor lages sikkerhetsnett, som ”Undo” og ”Revert to Saved”- kommandoer. Brukere må advares når de initiere en oppgave som vil forårsake irreversible tap av data. Hvis advarsler opptrer ofte kan det indikere noen designfeil. Vanlige problemer må forventes og brukere må gjøres oppmerksomme på mulige side-effekter. Utførlig tilbakemelding og kommunikasjon på hvert steg er derfor viktig slik at brukere kan ta de rette valgene.

3.3.9 Opplevd stabilitet

Apples retningslinjer definerer mange standard grafiske elementer som menylinjen og vinduskontroller som gir brukeren et kjent miljø hvor de vet hvordan ting oppfører seg og hva en kan gjøre med dem. For å gi brukeren en konseptuell følelse av stabilitet gir grensesnittet et klart, endelig sett av elementer og et sett handlinger som kan brukes på disse elementene.

3.3.10 Å ta hånd om kompleksitet i programvaren

Det beste utgangspunktet for å utvikle applikasjoner som er lett å bruke er å la designen være så enkel som mulig. Enkel design er god design, og de beste verktøyene er de som brukeren ikke engang er klar over at de bruker. Dess mer kompleks applikasjonens oppgave, jo mer viktig er det å holde grensesnittet enkelt og fokusert.

Progressiv avsløring er en teknikk for å redusere kompleksiteten ved å presentere det vanligste valget for bruker først, og så gi et valg som tillater brukeren å se tilleggsinformasjon og ytterligere valg. Denne teknikken gjør det lett for novise-brukere å forstå grensesnittet, mens det fortsatt gir erfarne brukere den avanserte funksjonaliteten de vil ha.

3.3.11 Oppsummering

I forhold til idealene angående intuitivitet ser vi at prinsippene ikke eksplisitt knyttes an til disse målene, men i introduksjonen til prinsippene er det skrevet: ” The implementation of Apple´s human interface principles make the Macintosh what it is: intuitive, friendly, elegant, and powerful” (Apple, 2008, s. 19). Vi ser altså at prinsippene danner de grunnleggende bygningsblokkene i streven mot idealene. Gjennom analysen vil det komme frem hvordan de ulike prinsippene understøtter intuitivitetsidealet sett fra et semiotisk perspektiv.

4. TEORI

4.1 Tverrfaglig fokus på forholdet mellom menneske og maskin

Det er gjort mye forskning på forholdet mellom menneske, maskin og organisatorisk kontekst (Andersen, 1990, s. 14). Disse emnene sorterer gjerne under forskningsfeltet Human – Computer Interaction (HCI), og er organisatorisk ofte underlagt informatikkfaget innen akademia. På Universitetet i Oslo tilbys det for eksempel et kurs i HCI ved Institutt for Informatikk. Erkjennelsen av at menneske-maskin forholdet kan belyses fra mange vinkler og innlemme innsikter fra ulike fag-grener har ført til at HCI-feltet har utviklet seg i mange retninger siden det grafiske brukergrensesnittets gjennombrudd, og dette reflekteres også i lærebøker og forskning (Preece et al. 2002, Kluge 2005). Jonathan Grudin (1990) skisserer i sin artikkel ”The computer reaches out: the historical continuity of interface design” hvordan perspektivet på grensesnittdesign har endret seg parallelt med datamaskinens utvikling, fra maskinvaren som grensesnitt for ingeniører på 1950-tallet til informatikeres interaksjon med programvare på 1960 og 1970 – tallet. 1970 til 1990 – tallet var dominert av terminalen, og perspektivet på grensesnittet skiftet til menneskelige faktorer, kognitiv psykologi og grafisk design. Fra 1980 – tallet fortsatte den kognitive psykologien og den kognitive vitenskapen å dominere som spesialdisipliner, og sluttbrukeren sto i sentrum av forskningen som en person i dialog med grensesnittet. 1990-tallet inkluderte flere disipliner i forskningen, som sosialpsykologi, antropologi og organisasjonsfag, da grensesnittet flyttet seg til arbeidsmiljøet og grupper av sluttbrukere. Ser vi på Apples retningslinjer er det klart at disse fant sin form i perioden 1970 – 1990 da fokus var på terminalen og sluttbrukeren. Selv om retningslinjene har forandret seg noe (blitt mer omfattende og detaljrike) er det slående likheter mellom 1987-utgaven og 2008-utgaven.

I tråd med HCI-feltets tverrfaglige nedslagsfelt vil jeg i denne oppgaven anlegge et semiotisk perspektiv på analysen av Dreamweaver. I hovedsak vil jeg benytte meg av Umberto Ecos semiotiske teori, i de neste avsnittene går jeg derfor gjennom hovedtrekkene i denne.

4.2 Semiotikk

I denne oppgaven har jeg valgt å fokusere på Umberto Ecos ”A theory of semiotics” (1979) som grunnleggende semiotisk teori. I det menneske – maskin interaksjon dreier seg om tegn og meningsrelaterte prosesser er det nødvendig med et teoretisk fundament som kan omfatte alle disse uttrykksformene på et tilpasset abstraksjonsnivå. Semiotikken er hensiktsmessig fordi den omfatter alle typer tegn i alle typer språk, som f.eks. tale, bilder, litteratur, film, teater og kroppsspråk (Andersen, 1990, s.3).

Semiotikkfeltet er bredt og ideene fortsatt heftig diskutert. Den følgende teoretiske utlegningen er verken en fullstendig summering av semiotikkfeltet eller noen enkeltstående eksisterende semiotisk teori. Kontroverser blant teoretikere innen fagfeltet er heller ikke tatt med, da utvalget er gjort først og fremst med henblikk på hva som er nyttig i forhold til problemstillingen i denne oppgaven. Det er et mål å bygge en teoretisk interpretasjon som tar sikte på å spleise semiotiske konsepter med etablerte konvensjoner i det eksisterende WIMP-paradigmet. Ideene til Umberto Eco bygger på sett og vis bro mellom ideene til semiotikkens ”fedre”, Ferdinand de Saussure og Charles Sanders Peirce, i det den semiotiske teorien integrerer det strukturelle og det kognitiv-interpretative perspektivet (Chandler, 2002, s.7).

4.3 Ecos kodeteori

Ved å beskrive brukergrensesnittet ut fra Ecos kodeperspektiv er det mulig å nyansere selve tegnbegrepet, forholdet mellom uttrykk og innhold samt de strukturelle forholdene tegnet inngår i. Samtidig omslutter interpretasjonsprosessen hele analysen, og denne vil naturlig nok knyttes til den prosessorienterte delen av Ecos teori. Med et slikt utgangspunkt håper jeg å kunne kaste lys over spenningsfeltet mellom domenespesifikke uttrykksbehov og arvede stilkonvensjoner innenfor WIMP-paradigmet. Samtidig er det viktig å presisere at idet analysen selvfølgelig innebærer nettopp interpretasjon, er den bærer av et visst subjektivt tidsstempel.

Generelt kan Ecos teori beskrives som to deler med en bro mellom. Første del beskriver en kodeteori; en generell utlegning av hvilke koder (signifikasjonssystem)

som ligger til grunn for kommunikasjon. Denne omhandler hvordan uttrykkselementer og innholdselementer er strukturert i forhold til hverandre på ulike plan, samt hvordan korrelasjonsregler kobler sammen uttrykksplanet og innholdsplanet. På grunnlag av denne koden kan så kommunikasjon foregå. Det vil si at et felles kodeunivers hos avsender og mottaker legger grunnlaget for utnyttelse av dette felles signifikasjonssystemet, og dermed utveksling av mening gjennom interpretasjon og produksjon av tegn. Den andre delen av teorien fokuserer på prosess, og Eco forsøker her å forklare hvordan tegnproduksjon og interpretasjon, og dermed kommunikasjon, utarter seg. Sett i forhold til min oppgave beskriver denne prosessorienterte delen av teorien forutsetningen for analysen i det denne nødvendigvis må involvere en personlig, tidsstemplet interpretasjon. Kodegrunnlaget er nødvendig for å forstå det grafiske brukergrensesnittet, samtidig som en beskrivelse av dette samme kodegrunnlaget er selve målsettingen med analysen. Kodegrunnlaget er altså både forutsetning for og resultat av interpretasjonsprosessen. De metodiske implikasjonene i forhold til dette paradokset blir diskutert i metodekapittelet.

Et signifikasjonssystem kan beskrives ved å se på de elementene dette systemet består av og hvordan disse er korrelert og strukturert. I kjernen av signifikasjonssystemet ligger *tegnfunksjonen* (sign-function), som i dagligtalen gjerne kun kalles et tegn. Jeg velger å benytte meg av betegnelsen tegnfunksjon i analysen for, i tråd med Eco, å understreke tegnets doble karakter. Denne doble karakteren er vesentlig ut fra et analytisk perspektiv på kodegrunnlaget. En tegnfunksjon kan ikke ha noe innhold uten å ha et uttrykk, og ikke noe uttrykk uten et innhold. Uttrykk og innhold forutsetter altså hverandre, og kalles gjerne tegnfunksjonens *funktiver*. Uttrykket til et tegn kan ha mer enn ett innhold, og innholdet til et tegn kan ha mer enn ett uttrykk. Funktivene i funksjonen kan være enkle eller sammensatte, altså diskrete tegn eller også en hel diskurs. Forutsetningen for å bruke begrepet tegnfunksjon er altså en allerede etablert korrelasjon mellom funktivene gjennom en sosial konvensjon.[6]

Den allerede etablerte korrelasjonen kalles *kode*. Slik er koden en forutsetning for tegnproduksjon og interpretasjon i det den lenker sammen uttrykk og innhold. Den legger grunnlaget for en felles forståelse av tegn, og dermed muligheten for

[6] Korrelasjonsbegrepet er bevisst valgt nettopp for å understreke funktivenes gjensidige avhengighet.

kommunikasjon. ”The theory of codes explains how one possesses rules of competence that permit one to disambiguate or to overambiguate, to form and to interpret given messages or texts” (Eco, 1979, s.129).

Uttrykksplanet er gjerne rammet inn av et sett *syntaktiske* regler for kombinasjon av uttrykk. Innholdsplanet er strukturert av et sett *semantiske* regler som regulerer innholdselementene, gjerne kalt *sememer*. Dette medfører at tegnfunksjonene ivaretar et dobbelt sett av relasjoner: ”.. every item in the code maintains a double set of relations, a *systematic* one with all the items of its own plane (content or expression) and a *signifying* one with one or more items from the correlated plane” (Eco, 1979, s.126). De systematiske relasjonene er etablert gjennom posisjoner og forskjeller som fremtrer når forskjellige fenomen er gjensidig sammenlignet med henvisning til det samme relasjonssystemet. Et slikt system av relasjoner dreier seg altså om abstrakte kombinatoriske muligheter (ibid, s.40). Det er viktig å presisere at bruken av ordet kode i denne oppgaven sikter til det totale kodeuniverset, altså systematiserte tegnfunksjoner. Det vil si at de syntaktiske reglene, de semantiske reglene og korrelasjonsreglene utgjør et udelelig hele som til sammen utgjør koden.

Enhver semantisk enhet kan bli et element i et semantisk felt (område). Idet semantiske enheter er kulturelle enheter, kan det semantiske feltet som en gitt semantisk enhet tilhører være et aspekt av verdensbildet til en bestemt kultur. Enhver kulturell enhet er et element i et system av andre kulturelle enheter som kan begrense eller videre definere dets mening, og enhver kulturell enhet kan bli et element i flere enn ett semantisk felt. **”**Fuzzy” konsepter kan være sememer som er åpne for ulike ”lesninger” basert på de ulike meningene de kan ha i ulike situasjoner, eller på grunn av de ulike måtene de kan kombineres med andre kulturelle enheter. Senere i kapittelet vil jeg komme tilbake til Ecos hypotese angående strukturen på det semantiske planet.

Et uttrykk (sign-vehicle) kan referere til et nettverk av posisjoner innen samme semantiske system, eller til et nettverk av posisjoner i forskjellige semantiske systemer. Disse posisjonene utgjør de *semantiske markørene* til et gitt semem, og kan være enten *denotative* eller *konnotative***.** Denotative markører er ikke avhengig av tidligere denotasjoner for å konstituere et semem, mens konnotative markører nettopp

er avhengig av foregående denotasjoner. Denotative markører utgjør det grunnleggende innholdet i et uttrykk, mens konnotative markører utgjør innholdet i en tegnfunksjon i det den støtter seg til en foregående denotasjon for å gi betydning til noe. Konnotasjoner avhenger derfor av at en tegnfunksjon danner utgangspunkt for, og derfor blir omgjort til et uttrykk (sign-vehicle), for et nytt semem. Denne evige kjeden av konnotasjoner er kjernen i det som kalles *uendelig semiose*.

4.4 Relasjonen mellom type, instans og kommunikasjon

Idéen om et kodesystem som premiss for kommunikasjon må relateres til den faktiske tegnproduksjonen og interpretasjonen, og dette gjøres ved å skille mellom uttrykkstyper (syntaktisk system) og uttrykksinstanser (produserte enheter) på uttrykksplanet, og innholdstyper (semantisk system) og innholdsinstanser (interpreterte enheter) på innholdsplanet. Tegnfunksjonen etablerer i første omgang korrelasjon mellom en abstrakt type i det syntaktiske systemet og en abstrakt type i det semantiske systemet, og de generelle typene danner så basis for produksjon av konkrete instanser ved tegnproduksjon og interpretasjon. Typebegrepet står sentralt i denne oppgaven, og det betegner de felles egenskapene som gjør klassifisering av instansene mulig. En uttrykkstype karakteriseres av felles egenskaper på uttrykksplanet, mens en innholdstype karakteriseres av felles egenskaper på innholdsplanet. Disse felles egenskapene på innholdsplanet, altså de felles semantiske markørene som definerer en innholdstype, kan være toposensitive eller ikke. Toposensitivitet refererer til egenskaper som ikke kan formes i ord, i alle fall ikke fullstendig, som for eksempel figurale eller vektorielle karaktertrekk (Eco, 1979, s.247). Men mange toposensitive egenskaper uttrykkes også i ord hvis det er etablert som vane, altså en kulturell konvensjon, og egenskapene dermed har nådd et visst abstraksjonsnivå. Eco gir eksempelet med ordet ”stopp”, og jeg tolker det dit hen at essensielle egenskaper ved relasjonen bevegelse/stillstand ikke egentlig fullstendig kan uttrykkes gjennom en statisk skriftlig uttrykksinstans. Andre egenskaper er ikke toposensitive og kan tilfeldig knyttes til et uttrykk uten at noe går tapt, som egenskapene til ”idé” eller ”semiotikk”. I tilfeller der egenskaper ikke kan uttrykkes ved hjelp av ord blir det umulig med det Eco kaller en metalingvistisk definisjon (ibid, s.247).

På neste side følger en skjematisk presentasjon av forholdet mellom de ulike begrepene brukt i ulike teoretiske kontekster: den formelle teoretiske modellen og kodeteorien som danner utgangspunkt for min beskrivelse av Dreamweavers signifikasjonssystem, og kommunikasjonsteorien som beskriver den faktiske utnyttelsen av signifikasjonssystemet og som dermed er selve grunnlaget for interpretasjonsprosessen. I min analyse, som konsentrerer seg om Dreamweavers signifikasjonssystem og grunnlag for forståelse, vil jeg benytte begrepene innholdsinstans, innholdstype (eller også semem), uttrykkstype og uttrykksinstans i beskrivelsen. Eco har en mer omfattende modell i sin bok; utdraget under er valgt fordi det representerer ordbruken og det teoretiske grunnlaget i denne oppgaven.

<table>
<tr><th colspan="2">Formell modell</th><th colspan="2">Kodeteori</th><th colspan="2">Kommunikasjons-teori</th></tr>
<tr><td colspan="2">Kontinuum</td><td colspan="2">Erfaring</td><td colspan="2">Kilde</td></tr>
<tr><td rowspan="4">Korrelasjon mellom funktiver</td><td>Posisjonerte enheter</td><td rowspan="2">Innhold</td><td>Interpreterte enheter (instanser)</td><td rowspan="4">Kode</td><td rowspan="2">Budskap</td><td rowspan="2">Mening</td></tr>
<tr><td>System av tomme posisjoner</td><td>Semantisk system (typer)</td></tr>
<tr><td>System av tomme posisjoner</td><td rowspan="2">Uttrykk</td><td>Syntaktisk system (typer)</td><td rowspan="2">Sign-vehicle</td></tr>
<tr><td>Posisjonerte enheter</td><td>Produserte enheter (instanser)</td></tr>
<tr><td colspan="2">Kontinuum</td><td colspan="2">Stoff ("stuff")</td><td colspan="2">Kanal</td></tr>
</table>

Figur 1, oversatt utdrag fra Ecos tabell 6 (Eco, 1979, s. 53). Viser relasjonene mellom tegnfunksjonene brukt i denne oppgaven.

4.5 Interpretasjon

Eksistensen av ulike signifikasjonssystemer vitner om at det finnes visse sosiale konvensjoner for tegnproduksjon og interpretasjon. Disse konvensjonene prøver Eco å

beskrive i sin teori om tegnproduksjon. Tanken er at ved å kartlegge de ulike prosessene som foregår ved tegnproduksjon av ulik art, vil en rekke spørsmål også kunne besvares angående signifikasjonssystemers utviklingsdynamikk. Særlig knytter dette seg til kreative produksjonsprosesser der ny kode foreslås og, avhengig av mottakernes interpretasjonsprosess, enten godkjennes og etableres gjennom sosial konvensjon, eller ikke forstås og dermed ikke blir del av et felles distribuert kodeunivers. I denne oppgaven er det interpretasjonsprosessen i forhold til Dreamweavers signifikasjonssystem som naturlig står i fokus. Men idet jeg faktisk skriver ned funnene, er jeg selv også deltaker i en produksjonsprosess.

Kodeteorien forklarer hvordan en viss kodekompetanse, kalt erfaring i tabell 1, gjør det mulig å forme og tolke gitte tekster, og hvordan ting kan gjøres entydig eller overtydlig. I forhold til Dreamweaver og fokuset på et ”intuitivt” brukergrensesnitt ligger det i kortene at brukergrensesnittet i størst mulig grad bør spille på konvensjonell, delt kode. Spørsmålet er i hvilken grad den jevne bruker kan forventes å ha kunnskap om Dreamweavers domene, og i hvilken grad det er mulig å benytte forventet delt kunnskap fra andre semantiske felt for å forklare funksjonaliteten.

Det eksisterer ofte et system av mulige *subkoder* og kontekstuelle valg og omstendigheter som koden til en viss grad forutser idet de konvensjonelt er knyttet til visse hendelser. Med henblikk på Dreamweavers grafiske brukergrensesnitt er det forutsatt (men ikke eksplisitt kodet) at brukeren forstår og velger et tolkningsalternativ ut fra det faktum at han/hun sitter ved en datamaskin og interagerer med en applikasjon. Tatt i betraktning at Dreamweaver også koster en del penger og at man oftest aktivt har installert applikasjonen, er det rimelig å anta at brukeren kjenner litt til bruksmuligheter og domenet.

I møte med ulike tekster kan ofte kodegrunnlaget være ukjent, eller det kan være vanskelig å foreta riktige kontekstuelle valg. Det er i slike tilfeller *abduksjon* gjør seg gjeldende. Abduksjon er en slutningsform Eco kaller ”the most daring of inferences”, og den benyttes oftest der uttrykksinstansen ennå ikke er kodet, er vagt kodet eller er i prosessen med å bli kodet (Eco, 1984, s. 39). En hypotetisk slutning foreslås angående et enkelt-tilfelle gitt en antatt regel og et resultat. Dette til forskjell fra *deduksjon* hvor et resultat deduseres fra en gitt regel og et tilfelle, som for eksempel i det den

underliggende programkoden til det grafiske brukergrensesnittet i Dreamweaver eksekveres og det grafiske brukergrensesnittet vises. Den tredje slutningsformen kalles *induksjon*, hvor en konkluderer med en regel gitt et tilfelle og et resultat. Abduksjon er altså basert på å gjette seg til kodegrunnlaget, ofte basert på visse omstendigheter. Det vil si en eksperimentell sporing av reglene bak et signifikasjonssystem som kan gi tegnuttrykkene der mening. Aduksjonsprosessen er første steg på veien til å berike kodegrunnlaget, da nye fortolkninger bidrar med nye korrelasjoner og dermed nye tegnfunksjoner.

Eco skiller mellom to former for abduksjon: *overkoding* og *underkoding,* hvor overkoding går fra eksisterende koder til mer analytiske subkoder, mens underkoding går fra ikke-eksisterende koder til potensielle koder. Begge prosessene er gjerne flettet sammen i både tegnproduksjon og interpretasjon. Kategorien *ekstrakoding* dekker disse tilfellene hvor begge prosessene opptrer samtidig og er vanskelig å skille. Ekstrakoding kan sees som prosesser i grenselandet mellom en kodeteori og en teori angående tegnproduksjon. “..overcoding and undercoding remain half-way between a theory of codes and a theory of sign production and interpretation,..”(Eco, 1979, s.137). Dette fordi prosessen med ekstrakoding benytter seg av både kjent kode som grunnlag for gjetningen, samtidig som prosessen ofte er opphav til kreative prosesser som fører til nye korrelasjoner og dermed ny kode.

En viktig implikasjon ved konseptet ekstrakoding er hvordan det knyttes til *diskursiv kompetanse.* Ofte forstår vi fraser og tekster og andre uttrykk uten særlig mental anstrengelse, det hender vi antar hva som vil bli sagt før det faktisk sies. Dette henger gjerne sammen med at vi allerede har erfart dem i liknende omstendigheter eller kontekster. Det underforståtte i mange utsagn avhenger av abduktive prosesser, og resonneringen hjelpes på vei av tidligere ekstrakoding som driver en til å velge den mest passende koden eller til å isolere en gitt subkode (der igjennom styres også valget av konnotasjoner). Stil- og sjangerbegrepene tar nettopp opp i seg denne idéen om et system av forventninger og en diskursiv kompetanse opparbeidet gjennom tidligere abduktive resonnementer og aktiv ekstrakoding. I denne oppgaven er det paradigmebegrepet som kan knyttes til diskursiv kompetanse og er opphav til ”instinktiv” forståelse i møte med nye applikasjoner. Budskapet, eller teksten, kan altså oppleves som en tom form som kan bli fylt med varierende mening basert på

synsvinkel og referanse til forskjellige systemer av konvensjoner. Den grunnleggende betydningen til en uttrykksinstans, denotasjonen, kan bli forstått slik senderen ønsker, men forskjellige konnotasjoner kan kobles til avhengig av mottakerens interpretative beslutning angående valg av semantisk bane. Dette vil jeg komme tilbake til i diskusjonen rundt hva som utgjør et uttrykksinstansens innhold.

I forbindelse med analysen av det grafiske brukergrensesnittet i Dreamweaver, hvor en maskin er på andre siden i kommunikasjonsprosessen og enhver respons er forhåndskodet i maskinkoden, er det nettopp den manglende evnen til ad hoc abduktive prosesser og ekstrakoding basert på visse omstendigheter som ofte er årsaken til kommunikativt sammenbrudd. Det ser ut til at det fokuseres på å bøte på denne manglende sensitiviteten i forhold til kontekst ved satsing på det som kalles semantiske brukergrensesnitt. Det vil si at grensesnittet fremhever informasjon- og bruksalternativer som antas å være relevante i den gjeldende brukssituasjonen (presentasjon av Windows 7 på dagen@ifi 30.oktober 2008)[7].

4.6 Ideenes leksikon

Eco foreslår, som en regulerende semiotisk hypotese, å modellere det semantiske universet (og dermed kulturen) i form av et leksikon (Encyclopedia). Et slikt leksikon eksisterer naturlig nok kun som en sammenslutning av alle de lokale temporære strukturene av kunnskap hos hver av oss. Men det globale semantiske leksikonet oppleves som reelt i det vi kommuniserer med andre gjennom valgte uttrykksinstanser og opplever å bli forstått. Ideenes distribuerte leksikon blir aldri fullbyrdet (takk og lov), men viser seg nyttig som regulerende idé når vi forsøker å forstå faktiske diskurser og tekster ved å konsultere den nødvendige delen av leksikonet (Eco, 1984, 79-86). Spørsmålet er; hvilken struktur (hvis noen) egner seg som hypotese for å ramme inn all kunnskap (erfaring)? Tatt i betraktning ideen om uendelig semiose, er det klart at en global sammenslutning av hvert enkelt uttrykks mulige kjede av kontekstavhengige denotasjoner og konnotasjoner nødvendigvis blir kompleks. Eco foreslår å strukturere ideenes leksikon som et nettverk av denotativer og konnotativer

[7] Presentasjon av Rune Zakariassen fra Microsoft Norge. Se forøvrig artikkel i digi.no: ”UiO-studenter fikk verdens første Windows 7-demo” http://www.digi.no/php/art.php?id=792444

modellert i form av et ”rhizom” (ibid, s. 81). Begrepet ”rhizom” er brukt metaforisk og hentet fra botanikken der det betegner en jordstengel. Brukt filosofisk har Eco hentet begrepet fra Deleuze og Guattari. Vesentlige kjennetegn ved rhizom-strukturen er at hvert punkt er knyttet til ethvert annet punkt, det er ingen innside eller utside, og strukturen er åpen og lett å modifisere. Ingen kan gi en fullstendig beskrivelse av hele rhizomet fordi det er multidimensjonalt og alltid i forandring (som kulturen selv).

Et vesentlig poeng er at i en struktur hvor hver node er knyttet til alle de andre nodene er det mulig med motsigende slutninger slik at de semantiske feltene til en kultur eller samfunn kan være enten utfyllende (complementary), motsigende (contradictory), eller likegyldig til hverandre. Rhizomet representerer et mangfold av interpretasjoner realisert i forskjellige kulturer, og det er uendelig fordi alle diskurser angående leksikonet sår tvil om den forrige strukturen på leksikonet.
Selv om en slik type leksikon, som et resultat av en sammenslutning, må være komplekst betyr det ikke at ikke strammere strukturert kunnskap eksisterer. Men slik strukturert kunnskap må nødvendigvis være lokal og flyktig og kan bli motsagt av alternative og like lokale kulturelle strukturer. Forsøk på å gjøre lokale strukturer unike og globale produserer nødvendigvis en ideologisk slagside (Eco, 1984, s. 81 - 84).

Ideen om semiosens univers (the universe of semiosis) i form av et rhizom vil av pragmatiske hensyn ikke bli aktivt brukt i analysen, men er tatt med av hensyn til visse metodiske utfordringer og begrensninger som vil bli tatt opp i metodekapittelet. Det semiotiske perspektivet i analysen vil heller bli understøttet av ordbokmetaforen som anvendelig strukturerende begrep. Eco demonstrerer overbevisende hvordan ordbokformatet teoretisk sett er uholdbart som generell semantisk modell, men at det like fullt kan være et nyttig pragmatisk verktøy i vår daglige interaksjon med omverdenen, da kontekst og omstendigheter gjør klassifisering mulig på *ad hoc*-vis. Det er selvfølgelig ikke mulig for meg å bruke annet enn min lokale *ad hoc* ordbok i analysen (eller oppgaven forøvrig). Men denne er relevant da den jo er en del av det globale semantiske universet.[8]

[8] Et ”bevis” på at jeg tar del i det globale semantiske universet er hvis oppgaven (med analysen) gir mening for leseren.

4.7 Den personlige ad hoc-ordboken

I den semantiske ordboken er sememene klassifisert i et treliknende system av overbegrep (hyperonymer) og underbegrep (hyponymer), og dermed er visse beslutninger tatt angående hvor et semem hører hjemme, selv om det selvfølgelig ”egentlig” kan passe i mange forgreninger. Ved interpretasjon av uttrykksinstanser etableres en kontekstsensitiv provisorisk ordbok hvor visse egenskaper ved sememene tas for gitt Jeg vet for eksempel at i møte med en mappe i det grafiske brukergrensesnittet til en hvilken som helst applikasjon kan jeg anta at det ikke finnes en absolutt og utvetydig representasjon av en mappe i mitt semantiske univers, men akkurat derfor må jeg, av pragmatiske årsaker, etablere en *ad hoc* ordbokliknende lokal representasjon. Denne er basert på kontekst og erfaring og vektlegger visse egenskaper som viktige, som for eksempel at den kan åpnes ved dobbelt museklikk, en egenskap jeg ikke ville antatt hadde jeg funnet en mappe i en fysisk arkivskuff. Jeg velger altså det mappealternativet (blant mange forgreninger) som ut fra erfaring og kontekst virker mest sannsynlig. Disse ordbokliknende arrangementene vi hele tiden stiller til rådighet er flyktige og pragmatisk nyttige hierarkiske revurderinger av det semantiske leksikonet som sikrer at vi forstår et uttrykk slik intensjonen er. Mange av de egenskapene vi tillegger overbegrepene i våre ordbokliknende trær er dypt forankret i verdensbildet til vår kultur, og noen egenskaper ved elementene er mer sentrale og motstandsdyktige enn andre gitt en ”normal” kontekst.[9]

Vi husker fra kapittel 4.3 at både innholdsplanet og uttrykksplanet i Ecos kodeteori består av instanser og typer. Slik jeg forstår typebegrepet kan dette sees i sammenheng med ordbokmetaforen som modell av innholdsplanet. I det en uttrykksinstans alltid må fortolkes for å kunne stå for noe, og siden uttrykkstypene er en del av det semantiske universet (jeg forstår det slik siden ingen uttrykkstype kan realiseres uten å bli en instans, de er en del av ”ideenes verden”), forstår jeg det slik at overbegrepene i Ecos ordbokmetafor (som likner en trestruktur), kan sammenlignes med det generelle typebegrepet. Det opprettes ved interpretasjon derfor et hierarkisk system av typer der visse egenskaper tas for gitt på hvert nivå (basert på kontekst og

[9] Fra et semiotisk ståsted kan for eksempel debatten angående ”snikislamifisering” av Norge (mars 2009) betraktes som en debatt angående hvor motstandsdyktig de sentrale egenskapene ved begrepet ”norsk” er i forhold til omdefinering.

omgivelser) noe som veileder semiosen mot den endelige interpreterte (forhåpentligvis riktige) innholdsinstansen.

I forhold til domenebegrepet kan dette relateres til et semantisk felt som av praktiske årsaker modelleres som en ad-hoc ordbok i trestruktur hos brukeren av Dreamweaver ved interaksjon.

4.8 Type/instans ratio: en måte å karakterisere forholdet mellom uttrykk og innhold på

Ideen om interpretasjon av uttrykksinstanser som en abduktiv prosess der varierende antall mulige veier kan følges i det evig foranderlige semantiske universet avhengig av kontekstuelle valg og gitte omstendigheter, gjør det fåfengt å forsøke å konstruere en *generell* typologi av tegnfunksjoner. En tegnfunksjon eksisterer som et resultat av menneskelig semiose influert av kode og kontekst både under produksjon og interpretasjon. Den prosessuelle karakteren gjør at det snarere er snakk om en karakteristikk av interpretasjonsmodier med grunnlag i en viss kontekst og basert på en viss kode, i dette tilfellet Dreamweavers signifikasjonssystem. Det er klart at en karakteristikk basert på min semiotiske prosess byr på metodiske utfordringer, og disse forutsetningene og begrensningene vil bli grundigere behandlet i metodekapittelet.

Analysen vil ikke kunne bidra med en utførlig karakteristikk av alle sider ved signifikasjonssystemet da det er flere perspektiv som kan anlegges i fortolkningen av et kodesystem. Min analyse bygges opp av to fokusområder der det ene dreier seg om strukturelle kjennetegn på uttrykksplanet, mens det andre tar for seg forholdet mellom uttrykk og innhold.

Jeg tar en deskriptiv tilnærming til de strukturelle og visuelle kjennetegnene, og Apples retningslinjer angående de standardelementene som brukes i Dreamweavers brukergrensesnitt samt det faktiske utseende utgjør de vesentligste holdepunktene i beskrivelsen. Analyseparameterne angående forholdet mellom uttrykk og innhold er forankret i Ecos prosessorienterte produksjonsteori, og relevante idéer derfra blir presentert i de neste avsnittene. Disse idéene brukes i analysen for å kunne si noe om

måten funksjonaliteten formidles på gjennom WIMP-paradigmets karakteristiske uttrykksformer (beskrevet i den strukturelle analysen).

For å karakterisere forholdet mellom uttrykk og innhold i brukergrensesnittet til Dreamweaver vil jeg benytte meg av Ecos konsept angående type/instans ratio (type/token ratio). Dette er en skala der en instans karakteriseres i forhold til om den utgår fra en uttrykkstype eller innholdstype. Relasjons-skalaen er teoretisk knyttet til tegnproduksjon, en prosess jeg forstår som å knytte uttrykk til et innhold. Man kan i produksjonsprosessen benytte allerede etablerte koder, eller komme opp med ny kode og dermed bidra til dynamikken i kodeuniverset.

Det er to slags ytterpunkter i en type/instans produksjonsskala: ratio facilis og ratio difficilis. *Ratio facilis* er en form (modus) for tegnproduksjon hvor en uttrykkstype er reprodusert av en uttrykksinstans (token). Dette gjelder for eksempel enhetene i skriftspråket; jeg kan reprodusere ordet ´facilis´ mange ganger i forskjellige fonter (*facilis*, **facilis**, . .), i forskjellig farger eller med pensel eller blyant, men ordet vil uansett være instans av den samme uttrykkstypen. En uttrykkstype karakteriseres av noen essensielle felles egenskaper på uttrykksplanet (for eksempel bokstaver med visse visuelle egenskaper i en viss rekkefølge) som gis mening i relasjon til andre uttrykkstyper.

Ratio difficilis er en form for tegnproduksjon hvor en innholdstype er korrelert med en uttrykksinstans. En innholdstype karakteriseres ved visse felles egenskaper på innholdsplanet. Det betyr, noe forenklet, at uttrykksinstansen er motivert av den semantiske typen; den projiserer visse semantiske markører. Dette er den type tegnfunksjoner Peirce kaller ikoner, altså uttrykk som ”likner” objektet. Begreper som ”likner” og ”motivert” må i tråd med Ecos teori der innhold er kulturelle enheter, forstås som forenklede måter å beskrive det konvensjonelle (og derfor kulturelt definerte) forholdet mellom kulturelt definerte semantiske markører i en innholdstype og en gitt uttrykksinstans. ”Likhet” (similitude) er motivert av visse innholdsmessige egenskaper, men samtidig basert på regler som fremhever noen parametre som relevante og ser bort fra andre som irrelevante. Regler (kode) er altså nødvendig for å oppdage motivasjonen både i produksjon og interpretasjon. Dermed er forholdet

mellom uttrykk og innhold aldri ”naturlig” eller ”instinktivt”, men tvert imot alltid kulturelt kodet (Eco, 1979, s.196). Dette poenget forklarer hvorfor intuitivitetsidealet avskrives og at jeg i oppgaven isteden fokuserer på grad av forståelse.

Siden min analyse i stor grad fokuserer nettopp på korrelasjonen mellom uttrykk og innhold er tegn/type ratioen relevant. Men begrepene facilis og difficilis er ikke like klart parallelt knyttet til forholdet mellom type og instans i interpretasjonsprosessen. Siden koden allerede er etablert er interpretasjonsprosessen enkel eller vanskelig, altså facilis eller difficilis, ut fra i hvilken grad fortolkerens eksisterende kodekunnskap overlapper den som er kodet inn i det aktuelle signifikasjonssystemet. Vi ser altså at vektlegging av interpretasjonsprosessen gjør det nødvendig med noen avgjørende modifiseringer i forhold til Ecos teori, men fokuset på korrelasjonsprosessen beholdes. Produksjon og interpretasjon er sammenvevde prosesser som benytter seg av felles kodegrunnlag der kommunikasjon oppstår. Prosessen med å etablere en korrelasjon mellom funktivene må derfor alltid ha i mente fortolkningsprosessen hvis det ligger en intensjon bak uttrykksinstansene, noe jeg forutsetter at det gjør. Derfor er det mulig å karakterisere koden ut fra utvalgte fortolkningsparametre; i første omgang altså forholdet mellom instans og type:

uttrykksinstans/ uttrykksinstans/
innholdstype ---------- uttrykkstype

Den ene enden av skalaen er merket uttrykksinstans/innholdstype, og her plasseres de semantisk motiverte uttrykksinstansene. I en abduksjonsprosess kan slike uttrykksinstanser referere til allerede kjente semantiske markører til en gitt innholdstype, og sammen med den gitte konteksten kan mottaker gjette betydningen og forståelse oppstå. Prosessen med å gå fra ikke-eksisterende koder til potensielle koder har jeg tidligere kalt underkoding. Vi ser altså at de semantisk motiverte uttrykksinstansene gir hint i den abduktive prosessen ved å bygge på delt semantisk

kunnskap mellom avsender og mottaker.[10] I analysen vil jeg vie de projiserte semantiske markørene oppmerksomhet i forhold til opplevd relevans og dermed forholdet til konvensjonelle regler for hvilke parametre som markerer likhet.

De klassiske elementene i WIMP-paradigmet (Windows, Icons, Menues, Pointing device) legger også til rette for aktiv overkoding, prosessen å gå fra eksisterende kode til mer analytisk subkode, da de byr på et overhengende rammeverk av allerede kodet kontekst og tegnfunksjoner. Ved hjelp av aktiv bruk av eksisterende kunnskap i form av underkoding og overkoding kan altså idealet om "det intuitive brukergrensesnittet" bli forsøkt realisert.

Type/instans-skalaen utgjør den ene aksen i analysen. En kartlegging på denne aksen vil kunne si noe om forholdene for abduktive slutninger; altså om det legges til rette for vågale hypotetiske gjetninger. Denne kartleggingen vil ikke føre til noen normative slutninger angående det grafiske brukergrensesnittet i Dreamweaver, men heller si noe om hvordan standardene angående uttrykksformer i grafiske brukergrensesnitt forholder seg til ulikt type innhold. Som vi har sett er det avgjørende elementet for forståelse et felles kodegrunnlag hos avsender og mottaker. Forholdet mellom type og instans er derfor viktigst når kodegrunnlaget er svakt, altså når korrelasjonen mellom uttrykk og innhold ikke er etablert fra før og det derfor ikke finnes et passende presist "oppslag" i ad-hoc ordboken angående den aktuelle uttrykksinstansen. Er uttrykksinstansen tilfeldig koblet til innholdet (uttrykksinstansen befinner seg helt til høyre på aksen), og jeg ikke kjenner til koblingen, er det nærmest umulig å etablere denne uten tilleggsinformasjon. Det vil si at opprettelsen av korrelasjonen krever det Eco kaller en metalingvistisk definisjon (Eco, 1979, s. 247). Mangfoldet av ekstradokumentasjon og kurs i ulike varianter vitner om at behovet for tilleggsinformasjon er stort.

[10] Spørsmålet er hvilke egenskaper som eventuelt er relevante og som dermed knyttes til en uttrykksinstans. Ofte kan vektlegging av visse egenskaper fremfor andre ha en verdimessig slagside. En kartlegging av projiserte semantiske markører ut fra et verdiperspektiv er utvilsomt interessant, men utenfor denne oppgavens interessefelt.

4.9 Lett å forstå/vanskelig å forstå: en måte å karakterisere forholdet til intuitivitetsidealet på

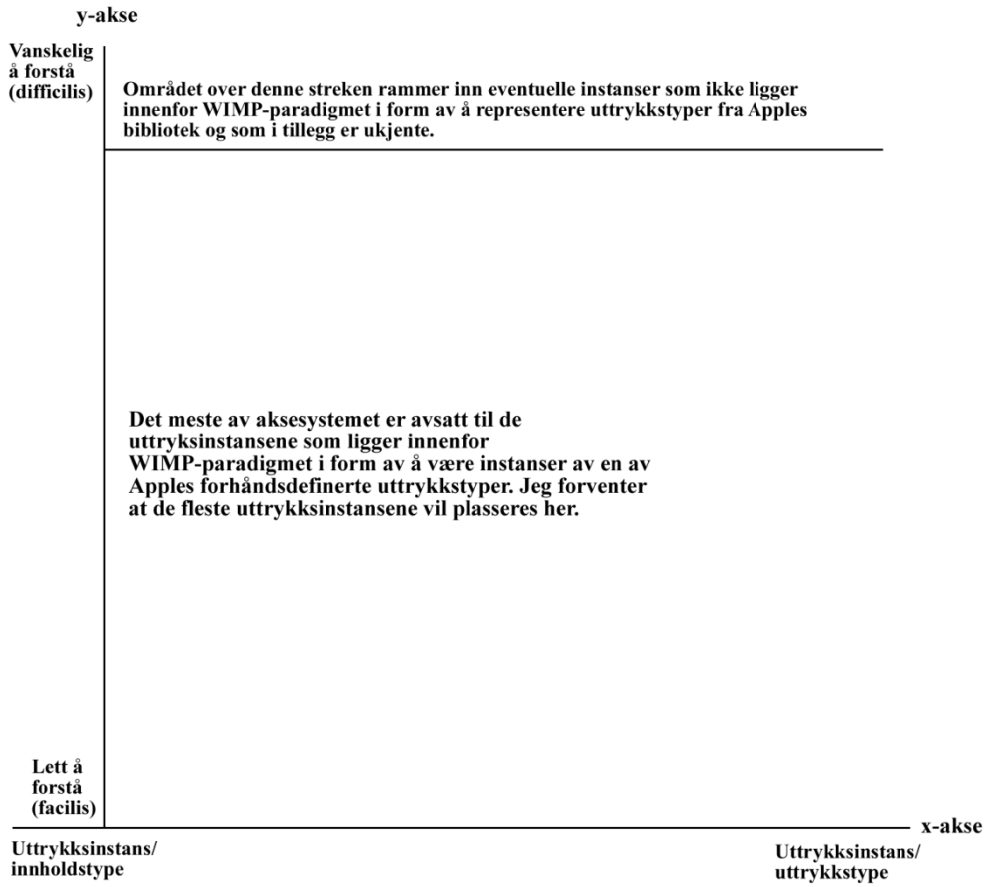
Den andre aksen i aksesystemet bygger på opplevelsen av delt kodegrunnlag. Det vil si hvorvidt jeg som fortolker av det grafiske brukergrensesnittet lykkes i å isolere senderens koder eller erstatte særegne subkoder med senderens. Lykkes jeg ikke, oppleves uttrykket som ren støy. Lett å forstå/vanskelig å forstå- dimensjonen i interpretasjonsprosessen kan sammenlignes med Ecos skala mellom facilis og difficilis i produksjonsprosessen. Det vil si at graden av overlapping i kodegrunnlaget på det aktuelle området mellom avsender og mottaker utgjør klassifiseringsgrunnlaget på skalaen mellom facilis (fullstendig overlapp) og difficilis (ingen overlapp). Facilis er dermed enkel kodegjenkjennelse (ved produksjon, eksekvering av kode), noe i nærheten av instinktiv (automatisk) bruk av tegnfunksjonen i det grafiske brukergrensesnittet. Dette er mulig gitt den etablerte konteksten og den begrensede ad-hoc ordboken jeg dermed etablerer. Å lære seg applikasjonen består dermed i å etablere denne ordboken slik at innholdet oppleves som logisk bundet sammen og alle uttrykksinstansene er representert som ”oppslagsord”. Ved ”instinktive” automatiske oppslag ender jeg altså i praksis den semiotiske prosessen og brukergrensesnittet oppleves som intuitivt. Jeg reflekterer ikke aktivt over betydningen av uttrykksinstanser men opplever uttrykk og innhold som en ”naturlig” enhet, som en tegnfunksjon. Slik oppleves også brukergrensesnittet som enkelt, naturlig og instinktivt etter hvert som kunnskapen økes, og det er vanskelig å tenke seg det annerledes. Grensesnittet blir ”usynlig” og applikasjonen gir seg selv slik tradisjonelle verktøy impliserer sin bruk.

Denne ideen om et praktisk stabilt kodegrunnlag som gjør det mulig å bruke applikasjonen hensiktsmessig og etter intensjonen, bekrefter ideen om et delt kodegrunnlag og er et metodisk forsvar og forutsetning for analysen. Muligheten for å komme til enighet om (eller kritisere saklig) uttrykksinstansenes denotative nivå (og som vi skal se også konnotative nivå) bekrefter en konvensjonell delt forståelse av signifikasjonssystemet og gjør karakteristikken av forholdet type/instans til noe mer enn en personlig tolkning.

Gitt definisjonen av intuitiv: ” umiddelbart oppfattende, som skyldes intuisjon og ikke erfaring eller refleksjon”, er det klart at begrepet i semiotisk perspektiv ikke kan brukes om uttrykksinstanser (Kunnskapsf. fr.ordbok, 2004). Som vi har sett har kodekompetanse alt å gjøre med erfaring og abduksjon (en form for refleksjon), altså omtrent det motsatte. Men siden begrepet er så sentralt i oppgaven, drister jeg meg til å definere ”intuitivt brukergrensesnitt” som et grensesnitt som ”gir seg selv”. På den måten innebærer begrepet en umiddelbar forståelse, og y-aksen representerer et graderingsskala i forhold til intuitivitetsidealet.

Gitt denne forståelsen av begrepet ”intuitivt brukergrensesnitt” er idealet innen WIMP-paradigmet at det *ikke* skal behøves opplæring utenfor grensesnittet selv; funktivene skal optimalt sett allerede være korrelert. På skalaen mellom full forståelse og null forståelse er det grader av forståelse basert på mulighetene for abduktive slutninger. Y-aksen gir derfor et bilde av om funktivene allerede er korrelert, om det er mer eller mindre mulig å korrelere dem gjennom abduktive slutninger, eller om uttrykket er fullstendig støy. Siden jeg stiller som Dreamweaver-novise er det interessant å kunne kartlegge hvordan lett å forstå/vanskelig å forstå-parameteret forholder seg til type/instans ratio og muligheten for givende abduktive prosesser, inkludert overkoding og underkoding. En mest mulig konvensjonell, stabil og omfattende ordbokseksjon på Dreamweaverfeltet kan stå som et slags pedagogisk siktemål ved bruk.

Kartleggingen av eget kodeunivers på et gitt tidspunkt i forhold til Dreamweavers signifikasjonssystem, altså forholdet mellom lett å forstå/vanskelig å forstå, er høyst personlig og tidsstemplet. Men min opplevelse av lett/vanskelig (facilis/difficilis) er eneste farbare vei, og antakelig kan noen karakteristikker også gjelde andre med liknende bakgrunn. På neste side vises de to aksene, samt hvordan WIMP-paradigmet som overhengende kjent kodekunnskap etablerer et grunnlag for abduktive prosesser.



Figur 2. Aksesystemet som brukes for å kartlegge forståelse og forholdet mellom uttrykk og innhold.

Mine muligheter for abduktive slutninger i forhold uttrykksinstansene i grensesnittet til Dreamweaver, er direkte avhengig av at jeg er i besittelse av kodekompetansen (erfaringen) som kjennskap til WIMP-paradigmet gir, men denne kompetansen er ikke tilstrekkelig for å oppnå full forståelse. Paradigmekompetansen er en generell kodekompetanse på abstrakt nivå (kan sammenlignes med sjangerbevissthet) som legger grunnlaget for visse forventninger hos brukeren av applikasjonen, og som er bevisst brukt av designerne av grensesnittet i Dreamweaver med det formål å gjøre dette mest mulig ”intuitivt”. I analysen tas altså WIMP-kompetansen som gitt.

I kontinuumet mellom fullstendig overlapp (som antakelig aldri forekommer) og ingen overlapp er det rom for abduktive prosesser og utvidelse av den overlappende koden. I abduksjonsprosessen er det gjerne flere forhold enn uttrykksinstansens utseende som spiller inn. Jeg har allerede nevnt strukturelle forhold og paradigmekompetansen, men i tillegg legger grensesnittet opp til ”prøv og feil”-

metoden ved å praktisere tilgivelse, og ved den tilleggsinformasjonen som tilbys av dynamiske uttrykksinstanser aktivert ved interaksjon. Aksesystemet jeg har presentert i dette kapittelet egner seg ikke for å kartlegge alle disse strategiene. Forståelse her begrenser seg til i hvilken grad jeg enten allerede kjenner tegnfunksjonen (uttrykk og innhold er en enhet i den gitte konteksten) eller kan trekke noen slutninger angående innholdet (funksjonaliteten) til den gitte uttrykksinstansen basert på hypoteser knyttet til projiserte semantiske markører i selve instansen og/eller ut fra relasjonene til omkringliggende uttrykksinstanser (eliminere kandidatfunksjonalitet og abstrahere felles kjennetegn).

4.10 Metaforer og analogier i semiotisk språkdrakt

Vi har sett at Apple anbefaler å bruke metaforer og analogier for å legge til rette for abduktive prosesser. Eco skriver at metaforer er ”nothing more than the substitution of one sememe for another, through the innovatory amalgamation of one or several markers. When the metaphor becomes customary, a *catachresis* takes place: two sememes acquire the same corresponding lexeme(..)”(Eco, 1979, s.109). En metafor bytter altså ett innholdselement med et annet ved oppfinnsom sammenstilling av en eller flere semantiske markører. Kontekst og omstendigheter er avgjørende for at uttrykk skal forstås i overført betydning, samt for å kunne trekke slutninger angående hvilke semantiske markører (innholdsmessige egenskaper) det er relevant å overføre. Det er viktig å understreke at metaforer gjør seg nytte av ulike, og ofte begrensede, semantiske markører ved den opprinnelige semantiske enheten. Mange metaforer er så innarbeidet i språket at de ikke lenger oppfattes som metaforer. Eco bruker begrepet katakrese om denne tilvenningen, altså at to sememer erverver samme uttrykk. Jeg vil betegne disse metaforene som døde metaforer, og de vil naturlig nok plasseres som kjente, semantisk motiverte tegnuttrykk i aksesystemet.[11] Et kriterium for å være død er å være blant de definerte grafiske elementene i Apples retningslinjer.

11 I Kunnskapsforlagets Norsk ordbok er katakrese definert som ”bruk av ord i strid med deres grunnbetydning” og ”bruk av innbyrdes uforenlige metaforer”. Dette er en smalere avgrensning av begrepet enn det jeg oppfatter at Eco opererer med, som går mer i retning av ideen om døde metafor.

Umberto Eco skriver om analogi at det er ".. a procedure instituting the basic conditions for a transformation" (Eco, 1979, s.201). Som en forlengelse av kritikken av Peirces ikonbegrep understreker Eco at det som eventuelt er i overensstemmelse, samsvar eller ligner hverandre er basert på kulturelle konvensjoner som operasjonelt startsted. En transformasjon er ikke noen naturlig overensstemmelse eller samsvar; det er heller en konsekvens av regler og påfunn. "Similitude is *produced* and must be *learned"* (ibid., s. 200). Bruk av analogier vil i likhet med metaforbruk plasseres på den enden av skalaen der innholdstype/uttrykksinstans befinner seg.

5. METODE

5.1 Metodiske implikasjoner

I kjølevannet til den semiotiske teorien følger det noen metodiske implikasjoner jeg vil beskrive i dette avsnittet. Avslutningsvis i metodekapittelet vil jeg spesifisere analysestrategien tydeligere.

Analyseprosessen er naturlig nok basert på interpretasjon og produksjon idet jeg skriver denne oppgaven. Slik sett er jeg i følge eget teoretisk grunnlag selv med på å forandre kodegrunnlaget. Metodisk sett har dette to problematiske konsekvenser; den ene er at jeg setter meg fore å fryse noe som i utgangspunktet er beskrevet som evig dynamisk; det andre er at jeg i forsøket selv bidrar til dynamikken. Eco skriver:

> Semiotics must proceed to isolate structures *as if* a definitive general structure existed; but to be able to do this one must assume that this global structure is a simply regulative hypothesis and that *every time a structure is described something occurs within the universe of signification which no longer makes it completely reliable.* (Eco, 1979, s. 129, original uthevelse)

Det at den semiotiske teorien ender opp med å peke på det flyktige ved sitt interessefelt kan ikke bety at dynamiske fenomen ikke lar seg beskrive. Konsekvensen er at det er nødvendig å understreke det partikulære ved analysen; den angår kun Dreamweaver og den er et resultat av min interpretasjon. Hva er så vitsen? Eco gir selv svaret i sin teori; siden både tegnproduksjon og interpretasjon innebærer semiose rettledet av gjeldende kodegrunnlag og kontekstuelle valg er min interpretasjon av tegnfunksjonen antakelig ikke fullstendig forskjellig fra andres. Det vil si, dette er en hypotese med det premiss at jeg er åpen om eventuelle flertydigheter i fortolkningen av brukergrensesnittet der jeg ikke klarer å bestemme, eller etablere, en entydig korrelasjon mellom uttrykk og innhold. Men tatt i betraktning den utstrakte tilleggsdokumentasjonen som beskriver Dreamweavers signifikasjonssystem, blir jeg særdeles overrasket om jeg finner uttrykksinstanser jeg ikke finner noen entydig forklaring på. Hvis fortolkningen er uakseptabel, altså at jeg bryter den allmenne kodekonvensjonen, er det enkelt å påvise og argumentere for at jeg tar feil. Dette bekrefter at det finnes en konvensjonell kode i Dreamweavers signifikasjonssystem.

Bruk av ordbokmetaforen forsvares ut fra praktiske hensyn. Men som nevnt er det kun rhizom-modellen som kan omfatte det dynamiske kodegrunnlaget og den uendelige semiosen. I forhold til Dreamweavers grensesnitt er innholdet til uttrykksinstansene ganske klart definert, rett og slett fordi det er knyttet til entydige algoritmer og fordi tvetydighet ikke lønner seg i forhold til brukervennlighet. Det vil si; jeg kan oppleve å ikke forstå uttrykksinstansene, men forholdene er lagt til rette (for eksempel gjennom omfattende dokumentasjon) for at jeg skal finne et entydig svar hvis jeg forsøker. Dette understrekes også av at grensesnittet formidler en begrenset mengde funksjonalitet fastlåst i gitte algoritmer. Det maskinelle kodegrunnlaget forandrer seg ikke, og de kausale årsakssammenhengene mellom grensesnittet som kontrollpanel og faktiske resultater i arbeidet man holder på med er uavhengig av, men ofte behjelpelig i, den abduktive prosessen. Det er selvfølgelig sannsynlig at jeg ikke knytter de samme konnotasjonene til uttrykksinstansene som andre brukere, men dette er mindre viktig i denne oppgaven fordi den tar sikte på å kartlegge den praktisk nyttige denotative betydningen, samt de testbare konnoterte interaksjonsformene. I sum betyr dette at Dreamweavers signifikasjonssystem i utgangspunktet er laget for å være mest mulig entydig i forhold til det praktiske formålet det tjener. Med veldefinerte begreper kan bruken av ordboksmetaforen forsvares.

5.2 Karakteristikk av meg som fortolker

En interpretasjonsprosess er alltid personavhengig, og analysen er naturlig nok preget av min erfaring og bakgrunnskunnskap. Jeg karakteriserer meg selv som novise i forhold til Dreamweaver, og har kun overfladisk kunnskap i forhold til webdesign. Når det gjelder generell kompetanse i forhold til datamaskinen er jeg antakelig bedre skolert enn gjennomsnittet, da jeg har en Cand.mag. grad i Informatikk fra Universitet i Oslo. Jeg har kjennskap til en rekke ulike applikasjoner, og har aktivt brukt min Macintosh siden år 2000. Dette gjør at jeg kjenner godt til implementeringen av WIMP-paradigmet gjennom Apples retningslinjer for design.

5.3 Gangen i analysen

Analysen vil ta utgangspunkt i det semiotiske kodeperspektivet slik det er beskrevet i teorikapittelet. Det vil si at analysen av signifikasjonssystemet (koden) til Dreamweaver vil følge ideen om at enhetene i koden opprettholder et dobbelt sett av relasjoner. På den ene siden en systematisk relasjon til alle enhetene på samme plan (uttrykksplan og innholdsplan), og på den andre siden et betydningsmessig forhold til en eller flere elementer fra det korrelerende planet. En konsekvens av dette teoretiske utgangspunktet er at en vellykket interpretasjonsprosess (og dermed et ”intuitivt” grensesnitt) forutsetter at det syntaktiske systemet oppleves som kjent og at koblingen mellom uttrykk og innhold enten allerede er kjent, eller at det gis nok hint gjennom strukturelle kjennetegn eller annen etablert kodekompetanse (som gjenkjennelse av projiserte semantiske markører) til at ekstrakoding er mulig og fører til riktig (ut fra den gitte konteksten) slutning. Det er klart at analysen innebærer interpretasjon, og den er dermed en praktisering av den prosessorienterte delen av Ecos semiotiske teori.

Basert på den teoretiske modellen over signifikasjonssystemet vil analysen ha to hovedfokus; det ene på de systematiske relasjonene grensesnittet opprettholder på uttrykksplanet, det andre på korrelasjonen mellom uttrykksplanet og innholdsplanet samt min opplevelse av å forstå eller ikke forstå uttrykksinstansene. Selve signifikasjonssystemet, altså Dreamweavers grafiske brukergrensesnitt, vil etter en gjennomgang av det overhengende struktureringsprinsippet slik det beskrives i Apples retningslinjer, deles inn i fire hovedelementer som etter tur vil analyseres ut fra de nevnte perspektivene.

I det følgende vil jeg gå mer i detaljer angående hvilke parametre de to hovedperspektivene innebefatter.

5.3.1 Det strukturelle perspektivet

Gjennom et strukturelt perspektiv på uttrykksplanet vil jeg forsøke å beskrive de syntaktiske reglene som eksisterer ved å kartlegge uttrykksinstansenes visuelle fremtoning og hvordan forholdene mellom dem etableres. En viktig målsetting med denne delen er å sortere ut viktige uttrykkstyper basert på felles karaktertrekk og romlig plassering. Gitt at Dreamweavers uttrykkstyper bygger på Apples bibliotek

over standard grafiske elementer, er visse føringer allerede lagt angående elementenes generelle funksjon, plassering på skjermen og relasjonene mellom dem.

En overlapping mellom Dreamweavers uttrykkstyper og Apples bibliotek over standard elementer legger grunnlaget for forståelse ved at viktige deler av den strukturelle koden på uttrykksplanet gjenkjennes og kan brukes i en overkodingsprosess (å gå fra eksisterende kode til mer analytisk subkode). Slik sett ser vi at en konformitet i forhold til standardene legger grunnlaget for realiseringen av intuitivitetsidealet.

Men gjenkjennelse av uttrykkstype og romlige relasjoner er kun et premiss og et steg på veien til faktisk forståelse, da det først er ved korrelasjon av uttrykksinstans og innholdsinstans at faktisk forståelse oppstår. Å gjenkjenne uttrykkstyper er derfor en nødvendig underliggende ”diskursiv kompetanse” som etableres gjennom WIMP-paradigmet, men det er eksistensen av en korrelasjon mellom de enkelte uttrykksinstansene og deres innhold (sememer) som til syvende og sist fører til forståelse.

Et interessant trekk ved grafiske brukergrensesnitt er at det til uttrykkstypene knytter seg interaksjonsformer. En interaksjonsform (for eksempel ”trykk”) er korrelert til forskjellige uttrykkstyper (for eksempel ikoner, knapper og kontroller). Samme innholdstype (interaksjonsform) er derfor korrelert til ulike uttrykksinstanser tilhørende ulike uttrykkstyper. I forhold til de to hovedperspektivene i analysen er det klart at en kartlegging av disse egentlig hører hjemme i delen angående korrelasjonen mellom uttrykk og innhold. Men tatt i betraktning at interaksjonsform er så ”instinktivt” knyttet til uttrykkstype og er lett å kartlegge (del av WIMP-kunnskapen og lett og utprøve), vil jeg av praktiske årsaker nøye meg med å nevne den tilknyttede interaksjonsformen i den strukturelle beskrivelsen.

Etter å ha beskrevet uttrykkstypene ut fra de gitte retningslinjene angående utseende og relasjoner (gitt av Apple og derfor en konkretisering av WIMP-paradigmets karakteristiske uttrykksformer), samt de visuelle kjennetegnene som springer ut av faktisk gruppetilhørighet i Dreamweaver, gjenstår de særegne visuelle karaktertrekkene som definerer en uttrykksinstans i forhold til, og som forskjellig fra,

alle andre. Disse er resultat av en produksjonsprosess slik Eco beskriver den i sin teori (innhold knyttes til uttrykk ved hjelp av eksisterende kode eller ved kreativ utprøving av nye korrelasjoner), men er samtidig underlagt konformitetspress i forhold til de visuelle egenskapene den deler med medlemmer i samme uttrykkstype. Instansene skal altså både tilhøre en gruppe og samtidig skille seg ut fra gruppen. De karakteristiske egenskapene er resultat av domenespesifikke uttrykksbehov, og de beskrives ved å fokusere på korrelasjonen mellom planene, altså forholdet mellom uttrykk og innhold. Det er altså i denne delen av analysen jeg forsøker å si noe om hvordan Dreamweavers funksjonalitet (domenespesifikke uttrykksbehov) formidles gjennom WIMP-paradigmets karakteristiske uttrykksformer (beskrevet i den strukturelle delen av analysen).

5.3.2 Y-aksen: grad av forståelse

Fokusområdet som omhandler korrelasjonen mellom uttrykk og innhold og graden av forståelse vil karakteriseres ut fra aksesystemet jeg presenterte i teorikapittelet. I første omgang vil jeg kartlegge forståelsen av uttrykksinstansene på y-aksen i koordinatsystemet. Jeg må ta for meg denne aksen først fordi kartleggingen her tar utgangspunkt i den kodekompetansen jeg besitter som novise. Det er et vesentlig poeng å få fram den semiotiske prosessen som eventuelt finner sted for å gjette uttrykksinstansens funksjonalitet ut fra strukturell plassering og uttrykksmessige kjennetegn. Alternativet er at instansen allerede er ferdigkodet i forhold til funksjonalitet (basert på tidligere erfaring) og at den derfor oppleves som ”intuitiv” (i det kodekompetanse og intuitivitet forveksles). Y-aksen i aksesystemet forsøker å fange inn kontinuumet mellom en diffus ”diskursiv kompetanse” og en spesifikk, pragmatisk, ferdigkodet forståelse av en spesiell funksjon i den gitte sammenhengen.

5.3.3 X-aksen: forholdet mellom uttrykk og innhold

Etter å ha plassert uttrykksinstansene på y-aksen vil jeg skifte fokus og konsentrere meg om korrelasjonen mellom uttrykksinstansene og det som i verktøymodus er det denotative innholdet, nemlig selve funksjonaliteten. Grunnen til dette er at de karakteristiske WIMP-uttrykksformene som er brukt kan karakteriseres ut fra hvilke forhold uttrykksinstansene har til det innholdet de representerer. Som vi skal se er mange av uttrykksinstansene i Dreamweaver mer eller mindre motivert av innholdstypen (gjennom projisering av stiliserte, kulturelt definerte vesentlige

egenskaper), og det er dette kontinuumet mellom fullstendig tilfeldig til semantisk motivert korrelasjon jeg forsøker å beskrive med x-aksen. Forholdet mellom type og instans har vesentlige implikasjoner for uttrykksformene i grensesnittet, og basert på tesen (som støttes av Apple) om at semantisk motiverte uttrykksinstanser kan gi grunnlag for vågale hypoteser angående funksjonalitet (i form av underkoding, prosessen å gå fra ikke-eksisterende kode til potensiell kode), er det interessant å finne ut om det er noen sammenheng mellom forståelse (og dermed intuitivitet) og forholdet mellom instans og type (karakteriserer uttrykksformen) i Dreamweavers uttrykksinstanser.

Plasseringen på x-aksen vil ta utgangspunk i en antakelse angående motivasjonen til de projiserte semantiske markørene. Denne antakelsen baseres på kunnskap angående den funksjonaliteten uttrykksinstansen skal formidle (hentet fra Dreamweavers brukerguide (Adobe, 2007)), kombinert med min personlige gjenkjennelse av hva ikonet skal forestille. Forholdet mellom innholdstypen(e) de semantiske motiverte uttrykksinstansene (ikonene) henter motivasjonen fra og det faktiske innholdet de skal formidle (funksjonaliteten beskrevet i brukerguiden), vil bli diskutert i analysen for å kaste lys over hvor godt uttrykksform og det faktiske innholdet (funksjonen) passer sammen (om funktivene er konvensjonelt kodet) i forhold til idealet om intuitivitet.[12]

5.3.4 Tilbakemelding og kommunikasjon ved interaksjon

Kartleggingen av uttrykksinstansene i koordinatsystemet forteller ikke den fulle sannhet angående forholdene for abduktive slutninger. Det er ofte lagt opp til valgfri tilleggsinformasjon i form av gule merkelapper tilknyttet den enkelte uttrykksinstans. Disse dukker opp når jeg fører musa over instansene, og jeg vil reflektere over deres eventuelle oppklarende funksjon etter at kartleggingen i aksesystemet er utført.

I tillegg til de gule merkelappene kan det være annen dynamisk tilleggsinformasjon eller reaktive tilbakemeldingsformer som benyttes av applikasjonen for å sikre brukerkontroll og forståelse. Disse vil bli påpekt der de oppfattes som relevante.

[12] For å forenkle refereringen vil ”brukerguiden” alltid henvise til Dreamweavers brukerguide (Adobe, 2007).

5.3.5 Pedagogikk versus profesjonalitet

I enkelte deler av analysen, der det er opplagt at en avveining er gjort i forhold til pedagogisk tilrettelegging og profesjonell bruk, vil det være naturlig å gi en kommentar angående dette.

5.3.6 Handlingstype og interaksjonsform

Det er avgjørende å ha distinksjonen angående handlingstype og interaksjonsform i mente i resten av oppgaven: eksplisitte og underforståtte handlinger representerer to forskjellige måter å kommunisere resultatet av og innholdet i en operasjon på. Altså handlinger der resultat og innhold er eksplisitt kommunisert gjennom uttrykksinstanser eller der innhold og resultat kommuniseres i sanntid i det den utføres (se Apples retningslinje 3.3.3). Bruk av begrepet interaksjonsform henviser til hvordan brukeren fysisk samhandler med maskinen; det kan være ved å dobbeltklikke, enkeltklikke, dra, taste inn bokstaver, tekstkommandoer osv.

5.4 Valg av analyseobjekt

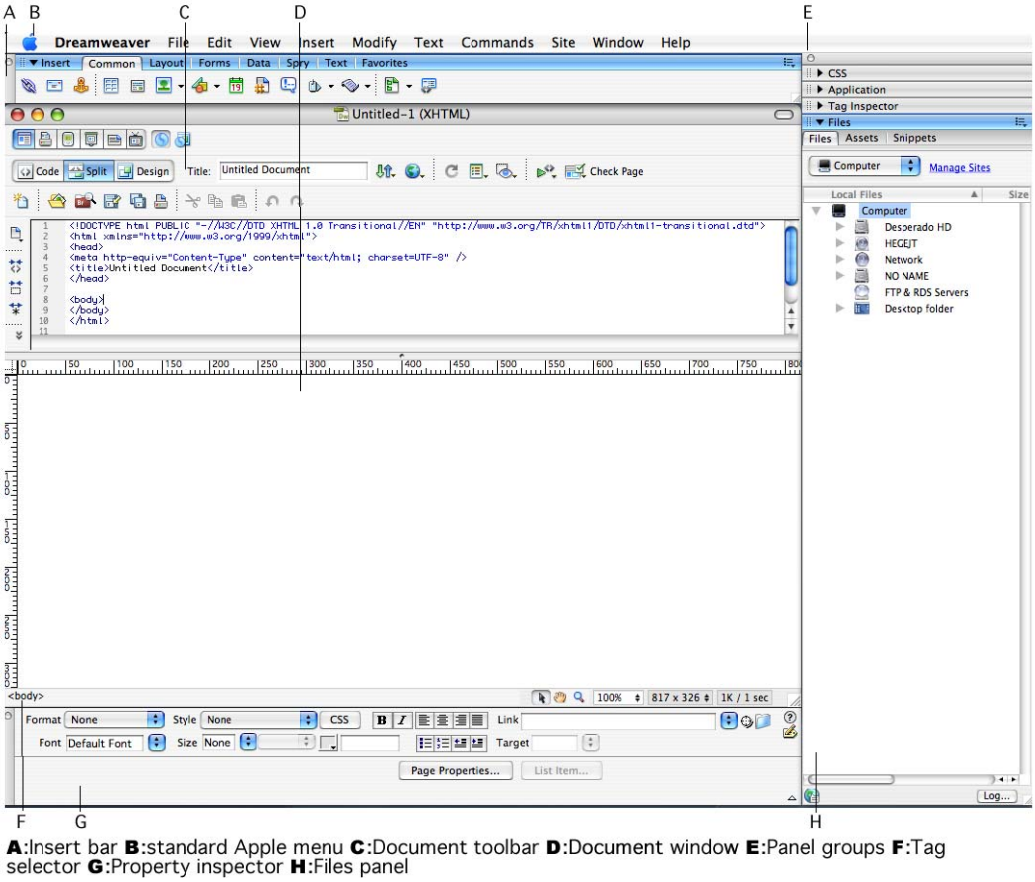
Jeg har valgt å konsentrere meg om fire hovedelementer i analysen. Dette er de brukerproduserte objektene, Dokumentverktøylinjen, Egenskapsinspektøren og Innsettingslinjen. En analyse av brukerproduserte objekter er essensiell fordi disse danner utgangspunkt for det meste av den andre funksjonaliteten i applikasjonen. Valget av de tre andre elementene er basert på noen antakelser angående deres innhold. Det vil si; de er valgt fordi jeg ønsker å analysere uttrykksinstanser som står for både domenespesifikk og ikke domenespesifikk funksjonalitet. I tillegg representerer disse tre elementene ulike typer av funksjonalitet, og til dels ulik uttrykksform.

6. ANALYSE

6.1 Syntaktiske og strukturelle kjennetegn i Dreamweavers grensesnitt

Under vises et utsnitt av Dreamweavers grafiske brukergrensesnitt der sentrale områder er gitt navn etter konvensjon fra Dreamweavers egen brukerguide. Dette utsnittet av Dreamweavers brukergrensesnitt viser standardinnstillingen med to ekstra verktøylinjer synlige og vil være utgangspunkt for analysen. Et raskt blikk på grensesnittet identifiserer fire hovedvinduer samt ikoner og menyer. I tillegg foregår interaksjonen ved hjelp av tastatur og mus; grensesnittet inneholder altså de vesentlige WIMP-elementene. De fire vinduene er standardelementer hentet fra Apples katalog, og de tre mindre vinduene rundt hovedvinduet kan betegnes panel.[13] De fire hovedvinduene har i oppgave å strukturere tegnfunksjonene innenfor sine rammer, og tegn med slike strukturerende oppgaver vil jeg kalle strukturerende tegnfunksjoner. Deres utstrekning i rom og beliggenhet i forhold til hverandre er avgjørende for forståelsen av de felles kjennetegnene som karakteriserer utvalget av tegnfunksjoner innenfor deres rammer, sammenhengen mellom de ulike elementene, samt styring av fokus. Vi ser derfor at størrelsesforhold, visuell omslutning i form av rammer, romlig utstrekning og plassering er sentrale egenskaper som er med på å bestemme den syntaktiske posisjonen i grensesnittet. Disse egenskapene spiller sammen med uttrykksinstansenes visuelle fremtoning for å forsterke tilhørigheten til en uttrykkstype. En slik tilhørighet legger grunnlaget for å gjenkjenne eventuelle delte sememer på innholdsplanet (felles konnotasjoner) og dermed gi grunnlag for en abduktiv prosess som kan fremme forståelsen av den enkelte tegnfunksjon og dens relasjon til andre tegnfunksjoner.

[13] Betegnelsen ”panel” kan brukes både på norsk og engelsk. Andre termer kan det være vanskelige å finne gode oversettelser på, eller det kan være lite hensiktsmessig i visse sammenhenger. Uansett følges konvensjonen med anførselstegn (””) der den engelske varianten er bevart.



Figur 3 viser navn på hovedelementene i grensesnittet til Dreamweaver

Før vi tar for oss et utvalg av de enkelte uttrykkstypene, både de strukturerende og de strukturerte, skal vi knytte den overhengende designen til et sentralt designprinsipp i Apples retningslinjer, nemlig det som tidligere er omtalt som: ”Reflekter brukerens mentale modell” (se kap. 3.3.1).

6.2 Overhengende struktureringsprinsipp: Reflektere brukerens mentale modell

Vi husker fra beskrivelsen av dette prinsippet (kap. 3.3.1) at det gir råd om hvordan et domene, altså et kunnskapsområde (semantisk felt), bør korreleres med et uttrykk ved å benytte allerede etablert kunnskap (kode) hos brukeren. Det vil si enten benytte brukerens eksisterende kode tilknyttet domenet, eller benytte eksisterende kode fra andre semantiske felt. Det betyr at hvis domenet ikke kan forventes å være kjent for bruker, bør det benyttes metaforer og analogier til allment kjente aktiviteter eller objekter for at brukergrensesnittet skal virke kjent (og dermed ”intuitivt”). Dette

kommer frem i det det heter at organiseringen av elementene i brukergrensesnittet bør reflektere brukerens forventninger angående komponentene og prosessene de inngår i eller representerer (Apple, 2008, s. 39-41). I semiotisk språkdrakt vil det si at relasjonene mellom sememene på det semantiske feltet gjenspeiles i systemet av uttrykksinstanser. Men i tillegg til forventninger angående tegnfunksjonenes relasjoner handler prinsippet om å kunne bygge på etablerte forventninger angående hvilke handlinger som kan utføres, samt forståelsen av de algoritmene de ulike uttrykksinstansene representerer. Denne forståelsen underbygges ved å benytte standard uttrykkstyper med konvensjonelt tilknyttede interaksjonsformer, samt gjennom de felles sememene (egenskaper på innholdsplanet) som danner grunnlag for en gitt gruppetilhørighet.

Uttrykksinstansene burde altså bygge på velkjente uttrykkstyper og inngå i en forhåndsdefinert struktur (modell), noe som vil innby til utforsking. Velkjente uttrykkstyper finner vi i Apples bibliotek over standard elementer og relasjonen mellom dem er i stor grad definert ut fra deres generelle funksjon. Dette gjelder særlig de strukturerende tegnene. For å oppsummere dette designprinsippet i forhold til kodeteorien kan det sies å gi råd om å korrelere allerede etablert strukturert innhold med standard uttrykkinstanser og syntaktisk kode gitt av WIMP-paradigmet gjennom Apples retningslinjer.

6.2.1 Dreamweaver og designprinsippet om mental modell

Spørsmålet som reises når tegnfunksjonene og relasjonene mellom dem i størst mulig grad burde gi seg selv, er om dette idealet alltid lar seg etterfølge, eller om det rett og slett finnes domener som vanskelig lar seg forklare med analogier eller metaforer. Som nevnt tidligere er valget av Dreamweaver som analyseobjekt i stor grad begrunnet ut fra ønsket om å finne ut hvordan funksjonaliteten innenfor et domene uløselig knyttet til datamaskinen som verktøy og medium, blir representert i brukergrensesnittet. Teorien om at brukerne av applikasjonen har relevante erfaringer fra liknende oppgaver i ”den virkelige verden” er derfor bevisst utfordret. Dette betyr ikke at jeg avskriver muligheten for slike analogier, men at jeg antar at disse er mindre opplagte og ”naturlige”. I forhold til bruk av analogier og metaforer i Dreamweaver må dette belyses fra to nivåer, det ene er på det overhengende konseptuelle makronivået, det andre fokuserer på de enkelte uttrykkstypene og

uttrykksinstansene. I det følgende vil jeg diskutere helheten i designen og hvilken mental modell den støtter. Kartleggingen av forholdet mellom type og instans vil kunne si noe om bruk av analogier og metaforer fra ”den virkelige verden” på mikronivå.

Definert som novise innen webdesign har jeg ingen forhåndsetablert mental modell av hvordan arbeidsprosessene innen webdesign utarter seg. Etableringen av en slik modell må derfor skje gjennom fortolkning av grensesnittet (gjennom abduktive prosesser hvis forholdene er lagt til rette for ekstrakoding) og eventuell tilleggsinformasjon (hvis intuitivitetsidealet svikter).

Tar vi en titt på grensesnittet til Dreamweaver (Figur 3) ligner dette på grensesnittet til andre applikasjoner som brukes for å produsere tekst (for eksempel Microsoft Word). De vesentligste kjennetegnene er et sentralt dokumentvindu med omkringliggende paneler og en hovedmeny øverst.[14] Kjennskap til denne strukturen og vinduenes funksjon gjør at jeg ved hjelp av overkoding kan gå fra eksisterende kodekompetanse (WIMP-paradigmet) til mer analytisk subkode i forhold til Dreamweavers grensesnitt. Jeg forstår at tekst, bilder og andre tegnfunksjoner kan plasseres i det hvite feltet, altså Dokumentvinduet, og at omkringliggende paneler antakelig er ressurser tilgjengelige for dette formålet. Oppgaven min består derfor i å samle, organisere og manipulere objekter til websiten. De strukturerende uttrykkstypenes sentrale funksjon i Dreamweaver, som at Dokumentvinduet viser brukerens objekter, mens panelene stiller med ressurser for manipulasjon og organisasjon, kan altså sies å implementere strukturelle og syntaktiske kjennetegn ved WIMP-paradigmet. Dette dokumenteres også i beskrivelsen av standardelementene i Apples retningslinjer for design (Apple, 2008). Men til tross for at jeg forstår uttrykkstypenes overhengende funksjon kan jeg ikke ved hjelp av abduksjon finne ut stegene i en naturlig arbeidsprosess (praktisk sekvens av handlinger i forhold til formål), og dette henger sannsynligvis sammen med mangelen på konseptuelle metaforer eller analogier som modellerer arbeidet.

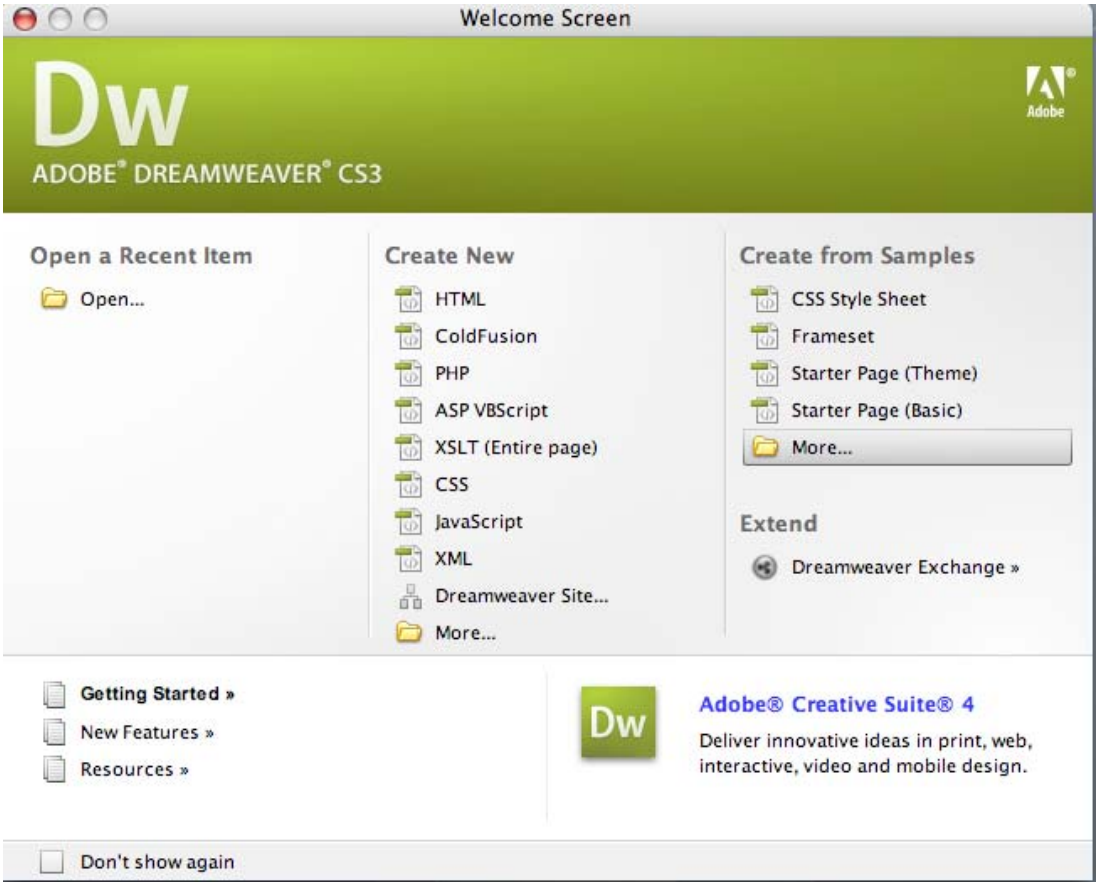
[14] Begrepet tekst brukes i vid forstand og omfatter alle typer av tegn i ulike sammenføyninger. Et slikt tekstbegrep rommer også det grafiske brukergrensesnittet til programvare, derfor forholder jeg meg til *en* type tekst for å produsere en annen når jeg anvender en applikasjon under tekstproduksjon.

Prinsippet om en mental modell er ment å understøtte en intuitiv bruk av applikasjonen ved at jeg skal kunne benytte allerede etablert kunnskap fra liknende arbeid. Vi ser altså at grensesnittet i Dreamweaver baserer seg på kjente WIMP-elementer men ikke gjør noen forsøk på å etablere noen overhengende konseptuell modell ut over dette. Det vil si, ofte er det brukt terminologi knyttet til før-digitalt arbeid for å forklare funksjonalitet og manipulasjonsmuligheter relatert til ulike typer av tekst (skrift, bilde, video o.l.). Begreper som ”Layout”, ”Font”, ”Edit”, ”Format” og liknende er arvet fra arbeid med tekst gjennom typografi, trykkerivirksomhet og liknende fra tiden før datamaskinen gjorde sitt inntog. I den grad det eksisterer en overhengende analogi er det altså den som er arvet gjennom WIMP-paradigmet og skrivebordet/kontorlandskapet som konseptuell metafor. I dag får de fleste kjennskap til terminologien knyttet til manipulasjon av ulike typer objekter nettopp gjennom applikasjoner på datamaskinen, det er derfor ofte ikke noe ferdig etablert semantisk felt (strukturert innhold) knyttet til begrepene. Designeren av Dreamweaver kan altså ikke forvente at brukeren allerede har en mental modell av arbeidet, og brukbare modelleringskonsepter fra ”den virkelige verden” er vanskelig å finne fordi datamaskinen i dag som regel er involvert i det meste av arbeid med tekst i vid forstand. I så måte kan det hevdes at den eksisterende overgangsanalogien (skrivebord, dokumenter, mapper og filer osv) har utspilt sin rolle som pedagogisk virkemiddel og blitt ”en død analogi” i tråd med det Eco kaller katakrese.

Dreamweavers velkomstvindu som sorteringsmekanisme

I balanseakten mellom pedagogiske virkemidler og det faktum at funksjonalitet i applikasjoner skal understøtte kreativ frihet i arbeidsprosessene, kan det være et poeng å unngå og legge for mange føringer gjennom designen av brukergrensesnittet i forhold til en gitt konseptuell modell. Tatt i betraktning at funksjonalitet skal brukes igjen og igjen er det viktige avveininger som må gjøres mellom pedagogiske virkemidler i brukergrensesnittet og profesjonell, effektiv bruk. Der det finnes opplagte mentale modeller å visualisere applikasjonens domene etter trenger det ikke være konflikt angående disse hensynene, men ser vi på det grafiske brukergrensesnittet i Dreamweaver ser det ut til at ansvarlig valgfrihet basert på etablert forhåndskunnskap om applikasjonen og webdesign generelt er prioritert fremfor pedagogisk rettledning.

For å bøte på, eller advare, om at visse forhåndskunnskaper er påkrevd for å kunne bruke Dreamweaver er det lagt inn en ”Welcome screen” som første stopp i det applikasjonen startes.



Figur 4 viser velkomstvinduet i Dreamweaver CS3

Som en slags første sorteringsmekanisme blir jeg implisitt bedt om å vurdere egne kunnskaper ved hjelp av en rekke veivalg inndelt i tre hovedkategorier. Jeg kan velge å fortsette et prosjekt, skape et nytt etter eget hode eller bruke ferdigsydde rammeverk. I tillegg til å være en inngangsport til selve applikasjonen er denne startsiden første steg på veien til opplæring. Ut fra hvilke kunnskaper jeg er i besittelse av er det mer eller mindre naturlige veier å gå. Normal vestlig leseretning tilsier at øynene først vil falle på kategorien ”Open a Recent Item”. For en ny bruker vil det naturlig nok ikke være noen tidligere prosjekter å fortsette med. Jeg vil derfor se på alternativene i neste kategori: ”Create New”, altså muligheten til å lage nye filer av ulikt filformat. De ulike valgene i denne kategorien kan virke nokså uforståelige for en bruker uten forkunnskaper innen webdesign. Valget består da i enten å la det stå til og velge det første og beste, eller begi seg videre til neste kategori for å sjekke

mulighetene der. Helt til høyre finnes kategorien ”Create from Samples”. I likhet med de uforståelige akronymene i forrige kategori gir valgene her lite mening for en fersk bruker. I det jeg begynner å forstå at jeg kanskje har tatt meg vann over hodet ved i det hele tatt å ha startet applikasjonen, finner jeg lenkene til ”Getting started”, ”New Features” og ”Resources” nederst til venstre. Jeg forstår ut fra startsiden at mine kunnskaper er mangelfulle og at jeg trenger en forklaring på en rekke begreper og sammenhengen mellom disse.

Velkomstsidens milde oppfordringen til et minimum av bakgrunnskunnskaper kan sees som et bevisst valg i forhold til det jeg tidligere har omtalt som teknologiens paradoks, nemlig at tekniske hjelpemidler blir så komplekse at de skaper flere problemer i forhold til bruk enn det de makter å hjelpe til med (Norman, 1988, s. 29). Men der Donald Norman og Apple hevder godt design kan gjøre bruksanvisninger overflødige og bruken intuitiv, forteller velkomstsiden i Dreamweaver at forkunnskaper er påkrevd for å kunne ha nytte av applikasjonen.

Etter å ha tatt innholdet i ”Getting started” nærmere i øyensyn forstår jeg at mye av arbeidet med websites faktisk begynner uavhengig av applikasjonen. Det vil si at det kreves forberedelse og en plan som applikasjonen så hjelper å sette ut i livet. Dette forarbeidet er det selvfølgelig vanskelig for applikasjonen å gi noen hint om, og en novise som meg har ingen mulighet til å vite at Dreamweaver ikke også kan være et planleggingsverktøy; akkurat som jeg skriver disposisjon i Microsoft Word. I brukerguiden gis det i tillegg hint om initierende prosesser og grunnleggende kunnskap som brukergrensesnittet i seg selv ikke avslører. For eksempel gis det grundig opplæring i hvordan sidene skal lagres i rotmapper både lokalt og på server, og en overhengende innføring i hvordan sites generelt er bygd opp. At en slik lagringsrutine er et naturlig første steg gis det ingen hint om hverken i velkomstvinduet eller applikasjonens grensesnitt ved oppstart. Tvert om er denne lenken plassert ganske usentralt i velkomstvinduet og godt gjemt i grensesnittet. Lagringsøvelsen er ikke del av en opplagt arbeidsprosess presentert gjennom designen, og jeg antar at det faktum at det kun er en initierende handling, det vil si at lagringsstrukturen gjerne kun etableres en gang, gjør at funksjonalitet ikke kan forsvare en for synlig plass. Altså en konkret avveining mellom profesjonell bruk, der den mest vanlige funksjonaliteten er mest tilgjengelig, og pedagogiske hensyn i

forhold til viktig, men mindre brukt funksjonalitet. Etter å ha lest om hvordan arbeidsforløpet burde foregå står det klart for meg at Dreamweaver er mer et eksekveringsverktøy for en noenlunde ferdig plan, og at struktur, layout og innhold planlegges uavhengig av selve applikasjonen. Dette optimale arbeidsforløpet og nødvendig kunnskap innen hvert steg formidles gjennom brukerguiden og ikke brukergrensesnittet, noe som vil si at brukergrensesnittet i seg selv ikke er velegnet til å gi en god mental modell av arbeidsprosessene. Som vi ser i forhold til de initierende prosessene som må utføres i Dreamweaver er det foretatt avveininger i forhold til hvilke valg som er vanlige og hvilke som er viktige. Dette er ikke alltid sammenfallende.

6.2.2 Oppsummering

Vi har sett at jeg ut fra mitt ståsted ikke kan gjenkjenne at Dreamweaver har modellert grensesnittet basert på noen åpenbare analogier ut over WIMP-paradigmet. Men en overhengende forståelse av sammenhengen mellom uttrykksinstansene oppstår fordi jeg kan gjette meg til forholdet i mellom dem ut fra plassering på skjermen. Denne gjenkjennelsen kan tilbakeskrives til Apples bibliotek av standardelementer (uttrykkstyper), deres standard syntaktiske posisjon og tilknyttet funksjonalitet. Gjenkjennelsen bygger rett og slett på forhåndskunnskaper om WIMP-paradigmet og Apples spesifisering av dette.[15] Slik ser vi at det generelle designprinsippet angående konsistens overholdes samt prinsippet angående opplevd stabilitet. Denne gjenkjennelsen av strukturelle egenskaper er et grunnleggende forutsetning for et ”intuitivt” brukergrensesnitt.

I neste avsnitt vil jeg ta for meg den sentrale strukturerende uttrykksformen vinduer. Først kommer en generell beskrivelse av egenskaper ved uttrykkstypen slik Apple definerer den (visuelle kjennetegn, funksjon, interaksjonsform og type tilbakemeldingsform), deretter kommer en kort presentasjon av Dokumentvinduet i Dreamweaver. Denne generelle beskrivelsen setter scenen for å kunne se nærmere på

[15] Apple tilbyr ulike verktøy og APIs (Application Programming Interface) for utviklere av programvare og tilhørende grensesnitt. ”Interface builder” er et verktøy med et standardisert bibliotek av grafiske grensesnittelementer som sikrer at applikasjonen under utvikling integreres i Mac OS X miljøet eller iPhone OS: http://developer.apple.com/documentation/UserExperience/index.html#//apple_ref/doc/uid/TP30000440-TP30000437

de viktigste uttrykkstypene i Dokumentvinduet, nemlig de brukerproduserte objektene.

6.3 Vinduer

Vinduer er en karakteristisk strukturerende uttrykksform i WIMP-paradigmet. De utgjør et standardisert rammeverk for å vise noe, bestemt av applikasjonen. En vesentlig egenskap er, i følge Apple, muligheten til å ha flere vinduer oppe samtidig, noe som gjør det lett å flytte og sette til side informasjon og samtidig ha den tilgjengelig gjennom overlappende vinduer. En annen viktig egenskap er muligheten til å manipulere vinduene ved å flytte dem, overlappe dem og forandre størrelse uten konsekvenser for innholdet, kun brukerens utsnitt. Denne egenskapen slutter seg til et sett konvensjoner for å åpne, lukke, flytte, manipulere størrelse, rulle (scrolling) og zoome vinduer.

Vinduene inneholder noen standardelementer som i stor grad definerer deres utseende. Et minimum er en tittelbar og en lukkeknapp. Dokumentvinduet har også gjerne horisontale og vertikale rullefelt (scroll bars), minimering- og zoomknapp, et proxyikon, tittelen på dokumentet, en ”resize”-knapp, en verktøylinjekontroll (toolbar control), samt gjerne også en bunnlinje (buttom bar).

6.3.1 Vinduer i Dreamweaver

I Dreamweaver er det i hovedsak fire vinduer, men som vi så i bildeutsnittet i begynnelsen av kapittelet har de fire vinduene ulikt utseende og oppførsel i tillegg til at de strukturerer ulik type informasjon og funksjonalitet. Dokumentvinduet (Document window) viser brukerdata, Innsettingslinjen (Insert bar) viser objekter som kan settes inn i dokumentvinduet, Panelgruppene (Panel Groups) viser kontroller og valgmuligheter, mens Egenskapsinspektøren (Property Inspector) egentlig er en subtype av panelfamilien som viser foranderlige egenskaper ved et valgt objekt. [16]

All denne informasjonen om navnene til vinduene samt deres funksjon er hentet fra Dreamweavers brukerguide, det er altså tilleggsinformasjon som ikke avsløres i grensesnittet selv. Kunnskap om dette må eventuelt tas med fra bruk av andre

[16] I tråd med Apple kan også Insettingsbaren defineres som en subtype av Panel.

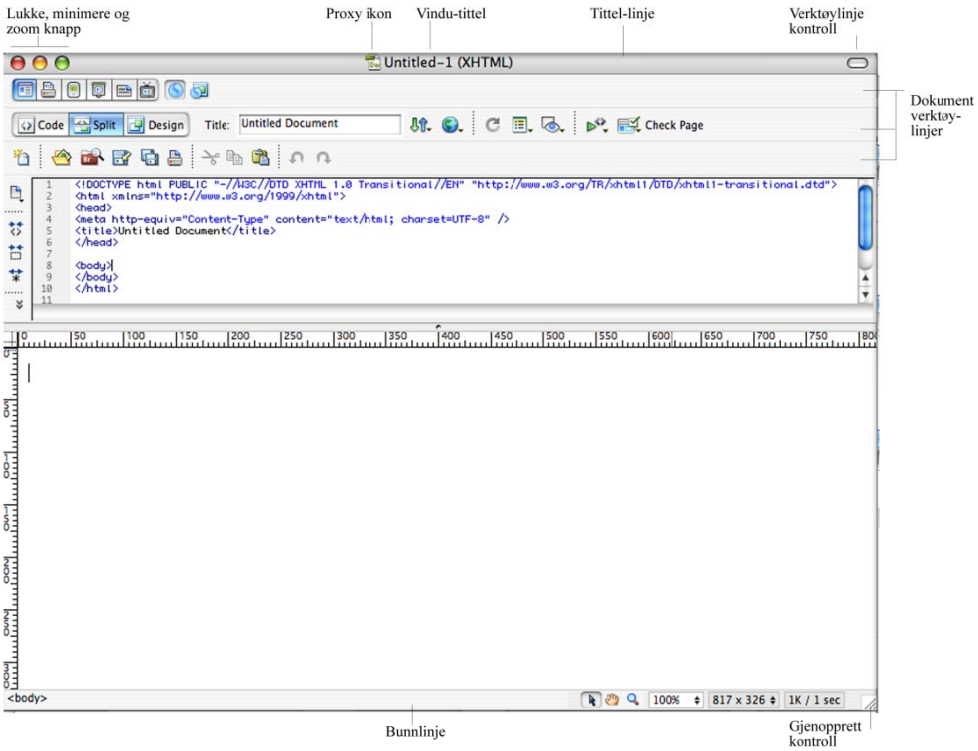
applikasjoner i det vi ser at vinduene er tilpasset Apples generelle retningslinjer og inneholder de standardegenskapene som er beskrevet over.

Vinduene er et eksempel på det generelle prinsippet om å bruke metaforer fra den virkelige verden i brukergrensesnittet. Det er også et eksempel på en død metafor som ikke gir mye tilleggsinformasjon til nye brukere.

I det neste avsnittet vil jeg ta for meg Dokumentvinduet i Dreamweaver. I omtalen av dette vil jeg gå nærmere inn på hva som kjennetegner de uttrykksinstansene det rammer inn.

6.3.2 Dokumentvinduet

Under vises et utsnitt av Dokumentvinduet. Som de fleste elementer i grensesnittet kan jeg velge ulike måter å vise dette vinduet på basert på mine preferanser. Under vises et splittet dokumentvindu (kode vises øverst, mens design vises under) der de fleste tilgjengelige verktøylinjer er inkludert.



Figur 5. Splittet Dokumentvindu med navn på standardelementene

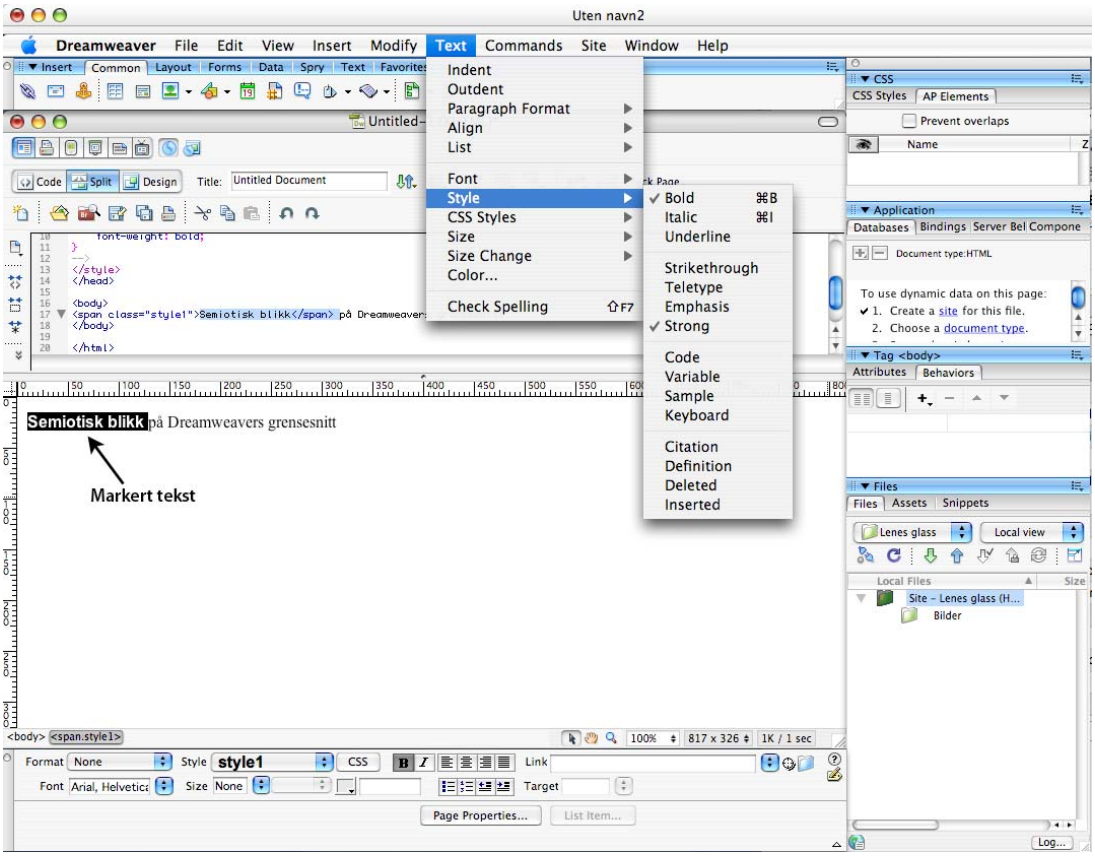
I Dreamweaver ser vi at det er Dokumentvinduet som er mest fleksibelt i forhold til størrelsesmanipulasjon. Det kan overlappes med vinduer fra andre applikasjoner, men Dreamweaver åpner selv kun nye dokumentvinduer i taber. Dokumentvinduet besitter standard vinduselementer (Figur 5) og kan manipuleres slik Apple foreskriver. Det rammer inn omfattende funksjonalitet hovedsakelig i form av ikoner, der hovedregelen ser ut til å være at ikoner som har noe til felles er plassert sammen i verktøylinjer. Den store hvite flaten er plassert i midten med en blinkende markør og påkaller dermed oppmerksomheten. Den øvre rammen der koden vises gir ut fra størrelse, nærhet til designvinduet, tekstlig innhold og det faktum at koden visuelt sett er svært forskjellig fra verktøylinjene rundt, grunnlag for en slutning om at innholdet kan manipuleres. I tillegg følger konvensjonen at sentrum av dokumentvinduer gjerne nettopp er for manipulasjon av egenproduserte objekter. Koden i seg selv gir antakelig ikke mening for en novise, men fordi kodeutsnittet oppdateres i tråd med designen i designvinduet er det lagt til rette for abduktive slutninger angående sammenhengen mellom de to vindusdelene under designprosessen. Jeg kjenner til grunnleggende HTML-kode og sammenhengen er derfor opplagt for meg.

Tatt i betraktning at de fleste uttrykksinstansene i grensesnittet forstås i relasjon til de egenproduserte objektene i Dokumentvinduet, er begrepet ”objekt” også særdeles velegnet syntaktisk sett. Et objekt er grammatisk sett en ”setningsdel som angir gjenstanden for en handling” (Kunnskapsf. fr.ordbok, 2004). Gitt at objektene ikke tilhører en generell uttrykksform (de er svært forskjellige avhengig av hvem som produserer dem), er det en karakteristikk av manipulasjonsmulighetene som står i sentrum i analysen av disse. I neste avsnitt vil jeg ta for meg objektene (gjennom et eksempel) og karakteristiske disse i form av det innholdet som står i sentrum i verktøymodus; nemlig tilknyttede handlingstyper og interaksjonsformer.

6.4 Interaksjonsformer og handlingstyper tilknyttet objekter i Dokumentvinduet

Som nevnt er sentrum av Dokumentvinduet avsatt til manipulasjon av egenproduserte tegnfunksjoner, såkalte objekter. Tilknyttet disse objektene er både eksplisitte og underforståtte handlinger, og disse handlingene utføres ved hjelp av en gitt interaksjonsform. Ved normal, formålsrettet bruk av applikasjonen er hovedfokus

gjerne på de egenproduserte objektene og funksjonalitet knyttet til disse. Uttrykksinstansene rundt fortolkes derfor gjerne som relevante eller irrelevante i forhold til intensjon og hvilke type objekt eller prosjekt man holder på med. Siden valget av interaksjonsform og fortolkningen av de eksplisitte eller underforståtte handlingene er avhengig av objektet som er markert, er det nødvendig å utforske interaksjonsformer og handlingstyper tilknyttet objekter gjennom konkrete eksempler. Under vises et bildeutsnitt der et tekstobjekt er markert.



Figur 6 med markert tekstobjekt

La oss gå gjennom handlingene som førte til eksempelteksten ”Semiotisk blikk” i bildeutsnittet over. Basert på kunnskap fra andre applikasjoner hadde jeg en idé om at jeg kunne skrive inn tekst der markøren blinket og at jeg for å manipulere teksten måtte markere den ved å klikke og dra musen. I det jeg utførte disse handlingene ble resultatet kommunisert ved hjelp av visuelle spor i sanntid; teksten dukket opp på skjermen i det jeg brukte tastaturet, og den fikk en svart ramme ved markering (se Figur 6). Ser vi på selve interaksjonsformene, altså inntasting ved hjelp av tastatur og markering ved hjelp av mus, er disse som alle interaksjonsformer tillært og gyldig på

tvers av applikasjoner som understøtter WIMP-paradigmet. Ser vi på hvordan resultatet av interaksjonen ble kommunisert, ser vi at jeg utførte en underforstått handling i det resultatet ble visualisert i sanntid. I bildeutsnittet over er det også gjort et menyvalg (en eksplisitt handling) direkte motivert av det markerte objektet. Valget av ”Text”-menyen er relatert til det markerte tekstobjektet og ønsket om å forandre tekstens stil (style). Men hvordan oppstår denne relasjonen mellom interaksjonsform, handlingstype og objekt? La oss belyse spørsmålet ved å se grundigere på hvordan forståelsen av eksempelobjektet ”Semiotisk blikk” etableres.

6.4.1 Objektenes doble innhold

Vi ser at det egenproduserte objektet ”Semiotisk blikk” har knyttet til seg flere interaksjonsformer og mangfoldig funksjonalitet av begge handlingstypene. Noe av svaret på hvordan relasjonene mellom et objekt, handlingstypene og interaksjonsform oppstår, tror jeg ligger i en aktiv omdefinering av objektets innhold i den gitte verktøykonteksten. I det jeg skrev ”Semiotisk blikk” tenkte jeg på ordenes ”normale” denotative betydning i lese- og skrivemodus. Når jeg så bestemte meg for å manipulere ordenes fremtoning, var innholdet i klassisk forstand totalt likegyldig i forhold til de handlingene jeg satte meg fore å utføre og de interaksjonsformene jeg visste var relevante. De egenproduserte tegnfunksjonene ble omgjort til objekter.

Teksten ”Semiotisk blikk” har altså minst to typer innhold knyttet til seg avhengig av kontekst; det ene har utspring i klassisk lesning og dreier seg om å forstå noe innenfor semiotiske rammer, den andre dreier seg om å forvandle ordene til ren tekstform, til et objekt klar for manipulasjon. Denne kontekstsensitiviteten har klare paralleller til den tidligere omtalte doble kommunikasjonsrollen der brukergrensesnittet veksler mellom å være kontrollpanel i ”verktøymodus” og klassisk medium i ”representasjonsmodus”. Jeg som hovedaktør i produksjonsprosessen (aktiv bruker av applikasjonen) veksler hele tiden mellom de to måtene å forstå objektene på. I det ene øyeblikket evaluerer jeg det klassiske innholdet som jeg ønsker å formidle med ordene ”Semiotisk blikk” når websiten er ferdig, i det neste vurderer jeg visuell fremtoning og mulige manipulasjonsmuligheter gitt av applikasjonens funksjonalitet. Vi ser derfor at What You See Is Whay You Get (WYSIWYG) – prinsippet spiller en viktig rolle i denne evalueringsprosessen.

Den kontekstuelle kodekunnskapen hos meg som bruker av applikasjonen hjelper meg med å etablere en egnet ad-hoc ordbok tilpasset det pragmatiske siktemålet jeg har med bruken av applikasjonen i øyeblikket. Det vil si, forståelsen av kontekst hjelper meg med å velge riktig innhold til uttrykksinstansene på skjermen fordi denne korrelasjonen allerede eksisterer (blant mange andre), eller den gjør det mulig for meg å etablere en ny korrelasjon mellom uttrykk og innhold ved hjelp av abduktiv resonnering. En viktig del av denne kontekstuelle kunnskapen er nettopp basert på praktisk kjennskap til WIMP-paradigmet slik det uttrykkes gjennom Apples retningslinjer.

Siden de underforståtte handlingene er avhengige av objektene som uttrykksinstans, analyseres disse i forbindelse med objekter. I neste avsnitt vil jeg se nærmere på hvordan jeg forstår resultatet av underforståtte handlinger og hvorfor disse oppleves som mer ”intuitive” enn de eksplisitte handlingene.

6.4.2 Underforståtte handlinger tilknyttet objekt

Objektet kan forstås som å representere en egen uttrykkstype i ”verktøymodus”. Essensielle kjennetegn er da det faktum at det er produsert av meg som bruker og at det er mulig å manipulere nettopp ved hjelp av funksjonalitet i applikasjonen. Alle objektene er tilknyttet et utvalg eksplisitte eller underforståtte handlinger, noen er felles, men en rekke varierer avhengig av hvilken type objekt som er markert. I eksempelet vi baserer oss på er objektet ”Semiotisk blikk” av typen ”egenprodusert skreven tekst”, og felles formelle kjennetegnene ved denne uttrykkstypen kan sies å være streker som utgjør distinkte bokstaver som stilles etter hverandre etter klare regler og som er tilfeldig knyttet til sitt innhold. Men viktigst i denne sammenheng er det faktum at innholdet i verktøymodus ikke utgår fra instansen, men selve uttrykkstypen. Tekstinstansen står kun som eksempel på uttrykkstypen, noe som i følge Eco medfører at type og instans faller sammen (Eco, 1979, s.226). Uttrykkstypen ”egenprodusert tekst” er altså representert ved tekstinstansen ”Semiotisk blikk”, men det er uttrykkstypen ”egenprodusert tekst” innholdet knyttes til. I det instans og type faktisk faller sammen, innholdet *er* uttrykkstypen i ”verktøymodus”, er det nettopp uttrykkstypen som er utgangspunkt for å forstå hvilke handlinger som relateres til det faktiske objektet. Spesielle egenskaper ved

uttrykksinstansen (objektet) blir da oversett fordi det er de felles typekjennetegnene og operasjonene knyttet til denne som er i fokus.

Vi har foreløpig kommet frem til at uttrykkstypen er den denotative betydningen av brukerens egenproduserte objekter i ”verktøymodus”. Ved at uttrykksinstansen blir ansett som en uttrykkstype som kan manipuleres, altså et objekt, representerer den en rekke handlinger som kan anses som konnotasjoner. De relaterte eksplisitte handlingene representeres ved egne uttrykksinstanser, de underforståtte handlingene har ikke noe eget statisk uttrykk, men i det de utføres gir de ”intuitivt” tilbakemelding om seg selv. Hva underbygger følelsen av intuitivitet?

For å finne ut hva som bringer frem den ”intuitive” forståelsen kan det være lurt å se etter hvilke egenskaper som er tilknyttet en underforstått handling: dette er tid, rom og bevegelse. En endret tilstand (bevegelse) knyttes alltid til tid (før og etter) og rom (manifestert virkning). Selve handlingen har ikke noe korrelert uttrykksinstans som relaterer de grunnleggende toposensitive egenskapene tid, rom og bevegelse i selve grensesnittet (dette måtte nødvendigvis være i form av video, animasjon eller liknende), men i det handlingen utføres er det en direkte korrelasjon mellom bevegelsen på skjermen og handlingen slik den er og blir forstått. Selve interaksjonsformen i underforståtte handlinger er alltid implisitt og må derfor læres uavhengig av grensesnittet. Vi har sett at bakgrunnskunnskap angående applikasjoner blir etablert ved at de følger samme norm, satt gjennom WIMP-paradigmet og presisert av Apple. La oss derfor se grundigere på Apples retningslinjer angående den underforståtte handlingen direkte manipulasjon.

6.4.3 Direkte manipulasjon

I følge retningslinjen angående direkte manipulasjon bør denne handlingstypen tilbys når brukere mest sannsynlig forventer det. Dette understreker nok en gang at etablert kodekunnskap angående passende interaksjonsform er avgjørende, og at denne interaksjonskunnskapen er resultat av gjentagende bruk av ulike applikasjoner innen samme paradigme. Jeg forventer for eksempel at hvis jeg markerer et tekstobjekt vil jeg kunne flytte dette ved å ”dra” i det med musen, fordi det er en vanlig underforstått handling på tvers av applikasjoner. Likeså forventer jeg at hvis jeg setter inn og markerer et bilde, vil jeg kunne manipulere størrelsen ved å dra i kantene; noe

Dreamweaver lar meg gjøre. Alle disse forventningene er utløst av tidligere erfaringer med andre applikasjoner og er ikke nødvendigvis knyttet til noen mental modell fra arbeid med tekst og bilder utenfor datamaskinen. Fysiske lover gjør tvert imot sitt til at markering av tekst eller bilde for så å flytte rundt i dokumentet eller dra i rammene slett ikke er mulig utenfor datamaskinens sfære.

Reglene for direkte manipulasjon er knyttet til de forskjellige objektene etter uttrykkstype. Det er for eksempel forskjellige handlinger knyttet til tekst og bilder; jeg kan dra i rammen til et bilde, men kan ikke dra i ytterkanten av den markerte teksten. Det å gjette hvilken interaksjonsform og handlingstyper som passer i en gitt sammenheng kan kalles overkoding: jeg går fra en generell eksisterende kode til en mer analytisk subkode gitt konkrete omstendigheter i Dreamweaver. Hadde jeg ved ren tilfeldighet kommet til å markere teksten ”Semiotisk tekst” for så og for eksempel dra i den ville jeg ved hjelp av underkoding kunne komme frem til potensiell handlingskode. Å ”oppdage” underforståtte handlinger er derfor prisgitt enten erfaring eller slump. Handlingsreglene antar jeg i de fleste tilfeller blir aktivt lært gjennom at noen demonstrerer handlingen, eller ved annen type dokumentasjon og opplæringsmateriell.

Videre stadfester Apples designprinsipp at en bør unngå å tvinge brukere til å benytte kontroller for å manipulere data. Det at den direkte manipulasjonsformen prioriteres kan kanskje forklares med at den gir umiddelbar tilbakemelding og dermed kontroll over og forståelse av prosessen. Idealet om det intuitive brukergrensesnittet blir derfor i større grad ivaretatt gjennom en slik interaksjonsform. Når det gjelder direkte manipulasjon må jeg kjenne til reglene for denne handlingstypen før jeg utfører den fordi den ikke avsløres av grensesnittet selv, som vi så er den en konnotasjon til den gitte uttrykkstypen (tekst, bilde, o.l.) i verktøymodus. Men, i det handlingen blir utført er det et direkte kausalt forhold mellom hendelsene på skjermen og det jeg foretar meg. Handlingen oppleves derfor som ”intuitiv” i sanntid. I det jeg drar det markerte tekstobjektet fra ett sted til et annet kan jeg se det flytter seg, jeg har fullstendig kontroll over prosessen, og jeg har en direkte forståelse av hvordan jeg kan reversere den. Det kan dermed virke som selve manipulasjonen og forståelsen av manipulasjonen er den samme; min forståelse av bevegelsen er lik den faktiske bevegelsen, og dermed er det ikke snakk om noen tegnfunksjon i det hele tatt. Og

denne tilsynelatende sammensmeltingen av uttrykk og innhold er det nærmeste en kommer det intuitive brukergrensesnitt. Mellomleddet er borte!

Men ser vi nærmere på manipulasjonsformen finner vi at fingerbevegelsen og uttrykksinstansens bevegelse er reell, men årsaksforholdet er simulert. Det er ikke min finger som faktisk flytter objektet; min fingerbevegelse setter i gang eksekveringen av en algoritme som får dette til å skje. Og dette er uavhengig av om jeg bruker mus eller fingeren direkte på skjermbildet (touchscreen). Derfor er det fortsatt snakk om representasjon ved hjelp av en bevegelig uttrykksinstans. Samtidig er det klart at jeg forstår manipulasjonsprosessen gjennom uttrykksinstansene på skjermen, så det som faktisk skjer inne i datamaskinen er uvesentlig i semiotisk sammenheng der fokus er på tegnfunksjoner. Uttrykket er dermed en hendelse på skjermen som projiserer tid, rom og bevegelse, og det faktum at disse toposensitive egenskapene er uttrykt basert på innholdstype og simulerer et kausalt avhengighetsforhold til min fingerbevegelse, gjør at den direkte manipulasjonsformen oppleves som naturlig. Men sett fra et semiotisk perspektiv der alle uttrykksinstanser er kulturelt kodet er det kjente (etablert kode) nok en gang forvekslet med det naturlige og intuitive.

Vi har i dette avsnittet angående de brukerproduserte objektene sett at det knytter seg et dobbelt innhold til disse. I ”representasjonsmodus” er objektet en uttrykksinstans som leses på ”klassisk” vis, det vil si at innholdet som knyttes til uttrykksinstansen er den som tilfaller ved ”normal” lesning. I ”verktøymodus” blir innholdet til objektet selve uttrykkstypen det utgår fra. Konnotert til dette denotative innholdet (tekst, bilde osv) er ulike handlingstyper og interaksjonsformer. Vi har tatt for oss den handlingstypen som er mest typisk knyttet til objektene, nemlig underforståtte handlinger. I neste kapittel vil jeg ta for meg de eksplisitte handlingene. Disse relateres ofte også til objektene i Dokumentvinduet fordi de eksplisitt tilbyr manipulasjonsmuligheter tilknyttet disse. Vi skal se at uttrykksinstansene som står for de eksplisitte handlingene også representerer en interaksjonsform i tillegg til en funksjon.

6.5 Eksplisitte handlinger

Jeg har frem til nå beskrevet de syntaktiske og strukturelle reglene i Dreamweaver sett i relasjon til det bakenforliggende WIMP-paradigmet. I tillegg har jeg analysert hvilke regler som ligger til grunn for de underforståtte handlingene og deres forhold til objektene. I dette avsnittet vil jeg ta for meg karakteristiske trekk ved korrelasjonen mellom uttrykksinstans og innhold tilknyttet de eksplisitte handlingene og tilhørende interaksjonsform.

De eksplisitte handlingene er nettopp eksplisitt uttrykt og har knyttet til seg en uttrykksinstans som skal stå for en operasjon. For å sette i gang denne operasjonen er uttrykksinstansen, gjennom den uttrykkstypen den tilhører, også knyttet til en konnotert interaksjonsform. Interpretasjonen av uttrykksinstansen med henblikk på dette doble innholdet er avhengig av brukerens kodekunnskap.

I neste avsnitt vil jeg innlede analysen av uttrykksinstansene til de eksplisitte handlingene i Dokumentverktøylinjen ved å ta for meg den strukturerende uttrykkstypen som rammer dem inn, nemlig verktøylinjen. Innenfor verktøylinjens rammer finnes ofte uttrykkstypen ikon, og denne vil også beskrives ut fra Apples dokumentasjon av de standard uttrykkstypene.[17] Jeg vil konsentrere meg om å gi en kortfattet beskrivelse av visuelle kjennetegn, tilknyttede interaksjonsformer samt uttrykkstypenes generelle oppgave i grensesnittet. Gitt abstraksjonsnivået er det først i analysen av de spesifikke uttrykksinstansene at jeg kan si noe om hvilke type innhold verktøylinjen rammer inn.

6.6 Verktøylinjer og ikoner

Verktøylinjer er en type vindu som skal gi brukeren umiddelbar tilgang til de mest brukte funksjonene i applikasjonen. Generelt skriver Apple om verktøylinjer at de skal utformes basert på brukerens mentale modell. Som nevnt har jeg som novise

[17] Verktøylinjer kan også ramme inn andre uttrykkstyper. Ikoner er tatt med her fordi det er en karakteristisk uttrykksform i WIMP-paradigmet, men i analysen av de konkrete verktøylinjene vil det bli gitt en beskrivelse av de andre uttrykkstypene som forekommer.

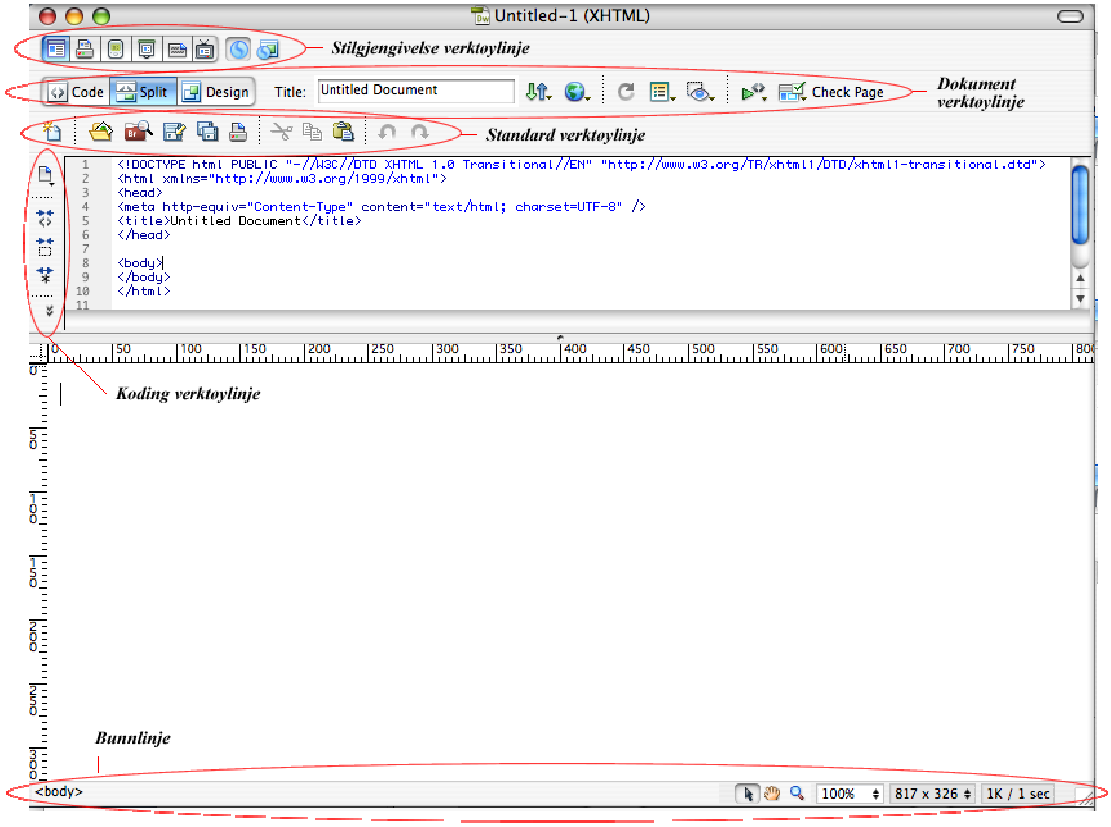
ingen etablert mental modell, så antakelig er verktøylinjen heller utformet etter hva designerne av Dreamweavers grensesnitt oppfatter som brukerens mentale modell (eller forveksler med egen modell). Rammene er mer eller mindre eksplisitt uttrykt, så viktigste definisjonsgrunnlag for verktøylinjen er den romlige tettheten mellom de ikonene den rammer inn. Vi kan derfor si at å strukturere ikoner (eller andre uttrykkstyper) er verktøylinjenes formål og eksistensgrunnlag.

Siden det er lite rom i en verktøylinje (Figur 7) må det prioriteres, og reglene som følges baseres på hva som er mest nyttig i forhold til den mentale modellen som ligger til grunn for designen. Det finnes ulike måter å arrangere uttrykksinstansene i en verktøylinje på, men generelt gjelder regelen om at de mest brukte, viktige, eller betydningsfulle instansene skal stå til venstre og ha størst synlighet. En hovedregel angående duplisering er at verktøylinjeinstansene skal ha en tilhørende menykommando, men det omvendte trenger ikke være tilfelle.

Ikoner er uttrykksinstanser som projiserer visse semantiske markører fra en innholdstype. I følge Apple representerer ikoner i verktøylinjer kommandoer som er hyppig i bruk i en applikasjon. De er gjerne små og mindre detaljrike og fotorealistiske enn andre typer ikoner, men som ikoner flest er de ofte velegnet til å representere toposensitive egenskaper som form, farge, posisjon, vinkel, størrelse, tekstur og mønster. Generelt gjør ikonenes visuelle egenskaper at de kan være kjappe å scanne med blikket samt lette å gjenkjenne og huske. Ikoner er i tillegg nyttige for de som ikke kan lese, og for at grensesnittet skal fungere globalt. Tatt i betraktning at grafiske designere gjerne arbeider og tenker i bilder kan ikoner være velegnet til å ”snakke deres språk” (Horton, 1994). I det følgende vil jeg ta for meg noen felles trekk ved verktøylinjene i Dreamweavers Dokumentvindu.

6.6.1 Felles trekk ved verktøylinjene i Dreamweavers dokumentvindu

Før jeg analyserer Dokumentverktøylinjen i Dreamweaver vil jeg påpeke noen felles egenskaper ved alle verktøylinjene. I bildeutsnittet under har jeg valgt å ha alle verktøylinjene synlige, men alle kan fjernes etter brukerens ønske bortsett fra bunnlinjen.



Figur 7. Dokumentvindu med navn på verktøylinjene.

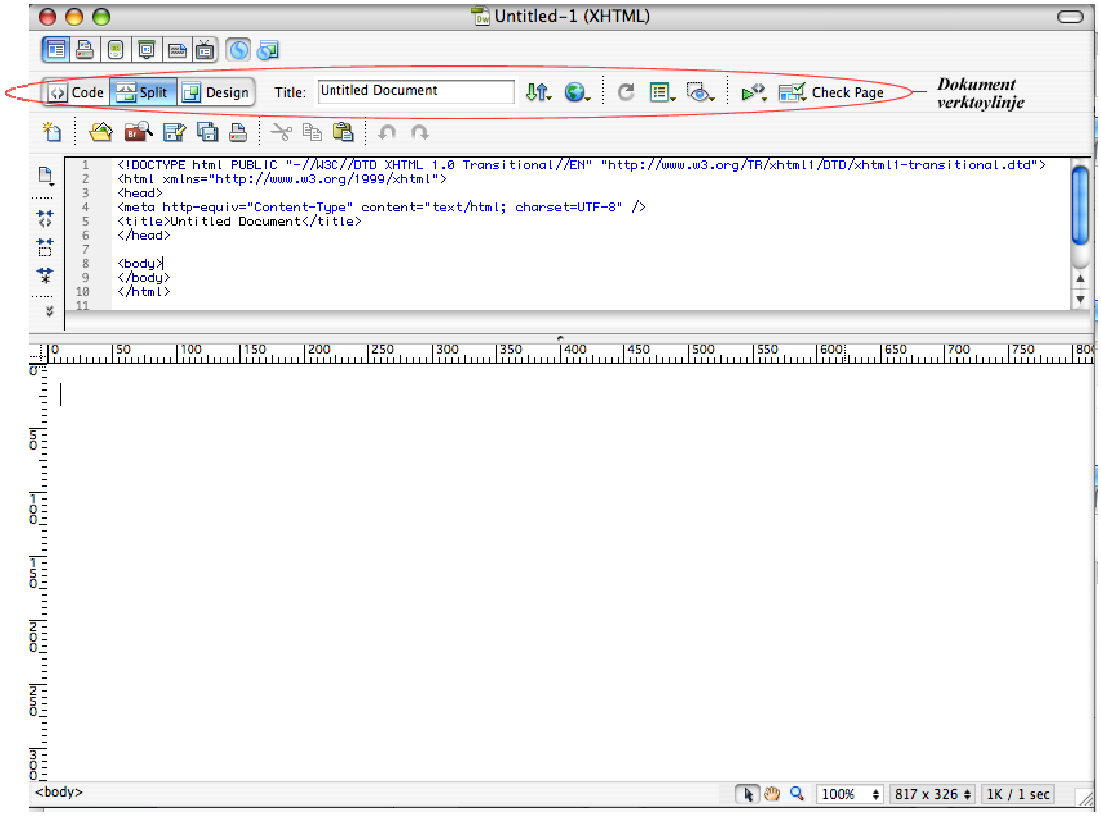
Et åpenbart trekk ved alle verktøylinjene er at de er uten synlige navn. For å finne navnet måtte jeg ta en titt i Dreamweavers brukerguide. Det er ofte en fruktbar strategi å sammenligne en tegnfunksjon i en verktøylinje med dem som naturlig er i samme familie for å gjette på felles vesentlige kjennetegn og dermed begrense tilfanget av mulig innhold. Gitt at ikonene er gruppert sammen av en grunn, nemlig de felles egenskapene, er det derfor noe underlig å ikke dette for brukeren og dermed være behjelpelig i den abduktive prosessen. Siden navnet på verktøylinjene ikke er synlig i grensesnittet vil det ikke regnes som etablert kodekompetanse i analysen eller trekkes inn som grunnlag for hypotetiske gjetninger angående innholdet til en uttrykksinstans.

En annen felles egenskap er at alle ikonene har tilknyttet en gul merkelapp som dukker opp hvis jeg fører musa over ikonet. Disse dynamiske visuelle egenskapene som defineres for visse uttrykkstyper utgjør dermed et typekjennetegn på uttrykksplanet og implementerer Apples regel angående tilbakemelding og kommunikasjon. Disse merkelappene kan være et nyttig hjelpemiddel i den abduktive prosessen hvis det tilbys tilleggsinformasjon, men kan også tyde på en erkjennelse om

at ikonene i seg selv kan være vanskelige å forstå intuitivt. En slik løsning med fleksible merkelapper er antakelig motivert ut fra den begrensede romlige plassen, og er mulig også en kompromissløsning mellom profesjonelle brukeres behov for et effektivt brukergrensesnitt og novisers behov for pedagogisk veiledning i form a tilleggsinformasjon. I selve kartleggingen av ikonene mellom ”lett å forstå” og ”vanskelig å forstå” vil jeg ikke ta hensyn til disse gule merkelappene i selve aksesystemet, men kommentere deres eventuelle nytteverdi i selve teksten. Dette fordi det er den samme uttrykksinstansen som bør vurderes i forhold til begge aksene.

6.7 Dokumentverktøylinjen

Under verktøylinjen for stilgjengivelse finner vi en lang verktøylinje kalt Dokumentverktøylinje (Document Toolbar). Jeg vil starte analysen med en beskrivelse av romlige relasjoner og visuelle kjennetegn samt identifiseringen av uttrykkstyper i denne verktøylinjen.



Figur 8. Dokumentverktøylinjen.

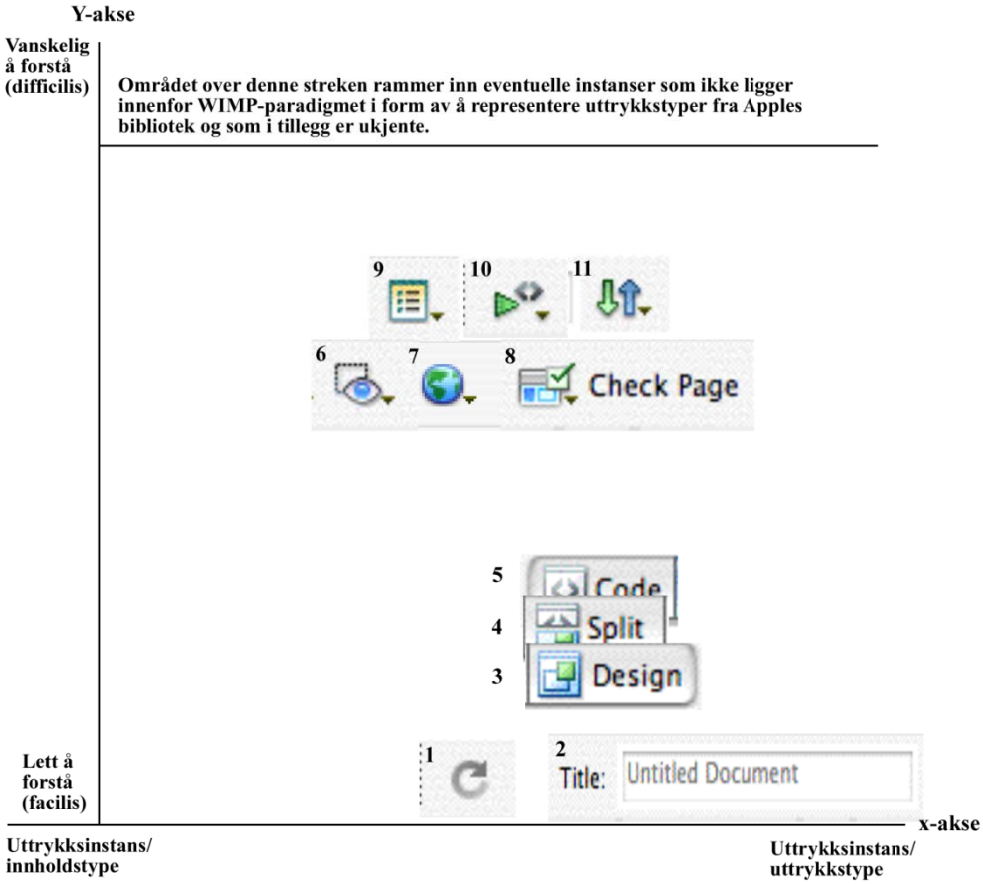
Vi ser at Dokumentverktøylinjen inneholder flere uttrykkstyper som ut fra visuelle kjennetegn kan deles i tre. Til venstre finnes tre sammenpakkede felt av typen segmentert kontroll (segmented control) med tre ekskluderende valgmuligheter. Kontroller er en subtype av ikoner som utløser umiddelbar handling eller synlige resultater når brukeren manipulerer dem med musa eller annen inputordning. I midten er det et input tekstfelt (Text Input Field) med merkelapp (Label), og til høyre en rekke firkantede ikoner av typen ikonknapper med pop-up menyer (Icon Buttons with pop-up menus). Tar vi for oss de segmenterte kontrollene skiller disse seg fra ikonene i de andre verktøylinjene ved at ett valg alltid er gjort, og dette ekskluderer de andre fra å forekomme samtidig. Det aktive valget markeres med blå bakgrunn som kontrasterer de andre feltenes bakgrunn. Når en kontroll blir valgt ved hjelp av interaksjonsformen ”trykk” forandrer Dokumentvinduet seg i tråd med valget, noe som gjør det enkelt å stille passende hypoteser angående deres funksjon basert på ”prøv og feil”-metoden. De tre segmenterte kontrollene er større enn de andre ikonene, mye fordi de inkluderer tekst i tillegg til bilde. Ser vi på input-tekstfeltet ser dette ut til å være innsunket i bakgrunnen ved hjelp av skyggelegging, og ved interaksjon gjennom ”trykk og input tekst ved hjelp av tastatur” blinker markøren i feltet og tekst kommer til syne. Ikonene til høyre er plassert ved siden av hverandre og av omtrent samme størrelse, men uten eksplisitte rammer. Ikonene med pop-up menyer skiller seg fra de andre ikonene ved å ha en konvensjonell liten pil nederst i høyre hjørne, og i det jeg trykker på et slikt ikon dukker det opp en liste med gjensidig ekskluderende valg der det aktive valget har kontrasterende blå bakgrunn. Vi ser også at ikonene på høyre side er delt i tre ved hjelp av to vagt stiplede vertikale linjer, såkalte separatorer (Separators). Lignende separatorer finnes også som vertikale skiller mellom verktøylinjene.

Vi ser ut fra den strukturelle analysen at Dokumentverktøylinjen implementerer Apples uttrykkstyper segmentert kontroll, input tekstfelt og ikonknapper med pop-up menyer. Dette innebærer at de beskrevne kjennetegnene er gitt, og at interaksjonsformen er etablert. I tillegg kommer standard oppførsel i form av visuell tilbakemelding ved interaksjon (som fargekoding) samt den romlige plasseringen som gir grunnlag for intern sammenligning. Gitt at jeg kjenner til Apples standard elementer er jeg allerede i besittelse av vesentlig kodekunnskap som utgjør en plattform for forståelse av den enkelte uttrykksinstans i verktøylinjen. I neste avsnitt

skal jeg ta for meg graden av forståelse i forhold til de enkelte uttrykksinstansene (y-aksen) samt karakterisere korrelasjonen mellom uttrykk og innhold (x-aksen) i tegnfunksjonene.

6.7.1 Aksesystemet

Tar vi for oss graden av forståelse samt forholdet mellom de enkelte uttrykksinstansene og det funksjonelle innholdet, vil kartleggingen av uttrykksinstansene i Dokumentverktøylinjen i aksesystemet se slik ut:



Figur 9. Aksesystemet der graden av forståelse og forholdet mellom uttrykk og innhold er illustrert.

6.7.2 Grad av forståelse

Tar vi for oss kartleggingen av forståelse først (y-aksen), ser vi at ikon nr. 1 og input tekstfeltet (nr.2) er plassert som enkle å forstå. Input tekstfeltet er enkelt fordi det går igjen i de fleste applikasjoner og er en velkjent uttrykkstype fra Apples bibliotek. Ikon nr. 2 er også gjenbrukt i flere applikasjoner og er et standard bilde Apple stiller til rådighet når funksjonaliteten som skal uttrykkes er ”Refresh the current view or

restart the process" (Apple, 2008, s. 154). Vi ser dermed at den funksjonaliteten som går igjen på tvers av applikasjoner (ikke domenespesifikk) standardiseres og bidrar til at intuitivitetsidealet føles nærmere. De tre segmenterte kontrollene (nr. 3, 4 og 5) er plassert som forholdsvis enkle å forstå med bakgrunn i den generelle antakelsen (gitt av WIMP-paradigmet) om at uttrykksinstansene i Dokumentvinduet representerer funksjonalitet knyttet til arbeidet her. Med den bakgrunnen er det mulig ved hjelp av overkoding å gjette hva teksten i kontrollene referer til. Sammenligner vi i tillegg kontrollene med hverandre og tar med i betraktningen at kombinasjonen av tekst og bilde forsterker hverandre, er det mulig å fremsette en sannsynlig hypotese angående funksjon. Det er en viss logikk i at "Split" er plassert i midten og innholder visuelle elementer fra begge de tilstøtende kontrollene, og selv med manglende kunnskap angående tags <> som en essensiell del av HTML-kode er det mulig å komme frem til en sannsynlig hypotese basert på teksten "Code". De fargelagte firkantene og teksten "Design" gir et siste hint i forhold til den semiotiske prosessen i tillegg til den vedvarende blå bakgrunnsfargen til valgte aktive kontroll.

Ikonknapp 6 er plassert som ganske vanskelig å forstå med bakgrunn i den generelle karakteren til informasjonen som gis. Øyet er et brukbart uttrykk for å uttrykke handlingen "å se" (få andre handlinger er tilknyttet øyet), og sammen med arket er det derfor mulig å fremsette en hypotese om at det dreier seg om funksjonalitet knyttet til å se på dokumentet. Men pragmatisk sett gir disse opplysningene ikke nok grunnlag til å vite hvilke muligheter som befinner seg i den tilhørende pop-up menyen.

Jordkloden (nr. 7) og ikonknappen med teksten "Check Page" (nr. 8) er stiliserte representasjoner jeg har vanskelig for å forstå fordi det konvensjonelt ikke knytter seg noe entydig innhold (i verktøymodus) til de projiserte semantiske markørene. Det vil si, bildet av jordkloden er vanlig å finne i grensesnitt og den står da gjerne for noe som har med Internett å gjøre. Jordkloden er et eksempel på bruk av metaforer, og jeg forstår at det er den semantiske markøren "utstrekning/størrelse" som blir byttet med tilsvarende semantiske markør i tegnfunksjonen "Internett". Det fusjonerte innholdet knyttes til det stiliserte bildet av jordkloden. Vanskeligheten oppstår når jeg i verktøymodus må knytte en handling til det stiliserte bildet. Det er ikke én konvensjonell handling knyttet til en jordklode, og selv med den gitte forutsetningen om at jordkloden er knyttet til dokumentet under bearbeiding finner jeg ikke noe

entydig svar. Jeg feiler altså i å spesifisere nøyaktig hvilken operasjon jordkloden representerer.

Ikonknappen med teksten ”Check Page” (nr. 8) er plassert som nokså vanskelig fordi bilde og tekst uttrykker nøyaktig samme innhold uten at dette er tilstrekkelig for full forståelse av funksjonaliteten. Det er vanskelig å vite hva siden skal sjekkes for.

I ikonknappene 9 og 10 er det svært vanskelig å kjenne igjen de stiliserte semantiske markørene. Nr. 9 kan likne en liste, og jeg kjenner igjen tagen i kontroll 10 fra kontroll 4 og 5. Men basert på dette er det vanskelig å komme med noen hypotese angående funksjonalitet. Verken liste eller tag har entydig handling knyttet til seg. Den ytterst stiliserte ikonknappen med pop-up meny og bilde av to piler (nr. 11) er plassert som lite forståelig da jeg kun aner at det må dreie seg om bevegelse av noe i to forskjellige retninger.

6.7.3 Forholdet mellom uttrykk og innhold

Tar vi for oss x-aksen og uttrykksinstansenes forhold til det definerte innholdet slik det kommer frem i Dreamweavers brukerguide ser vi at ikon nr. 1 har merkelappen ”Refresh Design View (F5)”, et innhold som er i tråd med den standard betydningen Apple har definert for bildet. Ikonet havner midt mellom de to ytterpunktene på x-aksen fordi det på stilisert vis projiserer visse semantiske markører ved at pilen indikerer bevegelse fra start til start, ikke for eksempel fra ett sted til et annet (Se forøvrig Eco angående vektorer, 1979, s. 240-241). Samtidig er det klart at sentrale egenskaper ved denne bevegelsen ikke lar seg projisere i et statisk ikon da de strengt tatt kun kan gjengis dynamisk (forandring av noe i tid). Den nærmeste gjengivelsen av disse dynamiske egenskapene er derfor den faktiske forandringen som skjer med objektene i Dokumentvinduet etter at algoritmen er eksekvert. Denne forandringen utgjør en form for visuell tilbakemelding som i større grad er semantisk motivert og dermed er nærmere å visualisere det toposensitive innholdet.

Input tekstfeltet (nr. 2) er tilfeldig koblet til sitt innhold og derfor instans av en uttrykkstype, mens de tre sammenvevde kontrollene (3,4 og 5) er plassert noe til høyre over midten av skalaen. Teksten i kontrollene er åpenbart hentet fra en uttrykkstype, mens de projiserte semantiske markørene visualiserer visse egenskaper

ved innholdet på svært stilisert vis. Kontrollen med tittelen ”Show Design View” viser to overlappende firkanter, men disse kan ikke sies å representere essensielle egenskaper ved konseptet design i seg selv (en design inneholder ikke nødvendigvis firkanter), men heller kalles et stilisert uttrykk for en design som semantisk er motivert av firkantformen. Et slikt uttrykk for design er nok valgt for å kontrastere motsatsen, nemlig den grå tagen som er visualisert i ”Show Code View”. Tags er et sentralt element i HTML-kode og bildet på kontrollen kan sies å være semantisk motivert av tagens toposensitive visuelle egenskaper. Den midtre kontrollen er tydelig motivert av selve dokumentvinduet ved at den visualiserer sentrale egenskaper ved et splittet Dokumentvindu, nemlig den vertikale linjen med en flik av kode fra kontroll 3 på oversiden og uttrykket for design fra kontroll 5 på undersiden. Et siste moment angående disse kontrollene er hvordan særlig det ikoniske uttrykket i kontroll 3 og 5 på hver sin måte er avhengige av ankerteksten (henholdsvis ”Design” og ”Code”) for å gi mening for en novise. Designkontrollen fordi det ikoniske uttrykket ikke inneholder noen karakteristiske projiserte semantiske markører som er kodet til et entydig innhold. Problemet er at ”design” er et abstrakt fenomen. I kodekontrollen er det på den annen side enklere å finne semantiske markører å projisere, men problemet her er at innholdet ikke kan forventes å være kjent av en novise. Plasseringen og synligheten til disse kontrollene tilsier at de er sentrale for alle brukere, og den relativt romslige plassen som er avsatt har den heldige effekten, i alle fall fra et pedagogisk perspektiv, at det er plass til gjensidig utdypende bruk av både tekst og bilde som uttrykksform innenfor kontrollens rammer.

Ikonknapp nr. 6 og 7 er plassert ved siden av hverandre på x-aksen, men det er ingen avgjørende forskjeller som tilsier at de ikke skal graderes likt, kun praktiske hensyn i forhold til plass. I ikonknapp 6 er det projisert semantiske markører fra et øye og en skjerm/dokument, og i forhold til handling (som er essensen i verktøymodus) er verbet ”å se” entydig knyttet til øyet. Det er derfor nærliggende å tro at det dreier seg om å se på et dokument eller liknende. Leser vi på merkelappen tilknyttet ikon-knappen ser vi at det står ”Visual Aids”, altså visuelle hjelpemidler. Øyet skal altså stå for et adjektiv, nemlig ”visuell”. Dette forklarer hvorfor jeg hadde vanskeligheter med å knytte en mulig funksjon til uttrykket; jeg er vant til å lete etter en handling. Firkanten er ikke behjelpelig med å føre semiosen i riktig retning idet den heller forsterker troen på at det har med ”å se” å gjøre ved å være et mulig mål for blikket.

Denne gangen ser vi altså at det byr på problemer å finne essensielle semantiske markører som lar seg projisere ved et adjektiv, kanskje fordi idéen ”visuell” i seg selv er så abstrakt at den paradoksalt nok ikke inneholder noen toposensitive visuelle egenskaper. Problemet i forhold til intuitivitet er dermed at de projiserte semantiske markørene som konvensjonelt koder ”øye” ikke kommer fra det sememet ikonet egentlig skal representere (visuell). Det andre stikkordet for funksjonen til ikonknapp 6, nemlig hjelpemiddel, blir ikke forsøkt uttrykt i det hele tatt. Dette er litt underlig tatt i betraktning den nærliggende tesen om at substantiv gjerne har semantiske markører som lar seg projisere. Men ”hjelpemiddel” er en nokså abstrakt idé som antakelig må forklares ved hjelp av eksempler, og gitt at ikonet er tilknyttet en pop-up meny er det forvirrende hvis egenskaper ved et konkret hjelpemiddel projiseres i menyhodet.

Ikonknapp nr. 7, med bilde av jorda, har knyttet til seg merkelappen ”Preview/Debug in Browser” og 77gir meg anledning til å forhåndsvise og korrigere i nettleser. Som nevnt i forbindelse med y-aksen er jorda en metafor, og ganske riktig har den med Internett å gjøre. Mer presist er den nok en metafor for Verdensveven (WWW), og da er jo nettleseren den mest kjente type applikasjon for å vise nettopp nettsider. Jorda som metafor kan altså ikke karakteriseres som tilfeldig knyttet til sitt innhold, men det er en kompleks semiotisk prosess å finne det sannsynlige innholdet. Handlingene ”å forhåndsvise” og ”å korrigere” er enda fjernere fra jorda som uttrykksinstans, men jeg antar handlingene er ment som konnotasjoner straks forbindelsen mellom jorda og nettlesere er opprettet. Tatt i betraktning at nettlesere har visuelle toposensitive egenskaper skulle en tro det var mulig å finne semantiske markører som lot seg projisere, men kanskje er det ingen distinkte egenskaper som klart definerer nettlesere som forskjellig fra andre firkantede uttrykksinstanser. Dessuten må ikonet være abstrakt nok til å fungere som et menyhode, og dette hindrer det i å projisere egenskaper gjennom konkrete eksempler. Handlingene ”forhåndsvise” og ”korrigere” er som andre handlinger knyttet til forandring i tid. Dette er (som nevnt tidligere) vanskelig å visualisere ikonisk så lenge elementene som inngår ikke har semantiske markører som lar seg projisere statisk. I tillegg kommer i dette tilfellet at eksempler er avskrevet på grunn av instansens funksjon som menyhode.

Ikonknapp nr. 8 med teksten ”Check Page” er plassert omtrent likt som de tre segmenterte kontrollene (3,4 og 5) med bakgrunn i at den også uttrykker både tekst og bilde. Teksten ”Check Page” gjør det lett å forankre bildet og dermed se hva som motiverer det. Vi ser at ikonknappen delvis er semantisk motivert i det den største firkanten projiserer noen karakteristiske egenskaper ved en nettside, som toppbanner og tabellformat. Bildet av en sjekkboks knyttes til handlingen å sjekke, og den er en interessant løsning på det problemet vi stadig kommer over når det gjelder å finne egnede semantiske markører som kan projisere handling (verb). Den viser ingen egenskaper ved handlingen, men derimot et eksempel på hvordan et resultat av handlingen noen ganger ser ut. Siden sjekkboksen er et av de definerte elementene i Apples bibliotek og ofte vises nettopp i forbindelse med handlingen ”å sjekke”, kan uttrykksinstansen forventes å være kjent, men den kan derfor også misforstås slik at det ser ut som en side faktisk skal sjekkes ved hjelp av en sjekkliste. I kombinasjon gir teksten og bildet ikke noen utfyllende informasjon ut over nettopp ”check page”, og dette er som nevnt ikke nok for at jeg skal forstå hvilken funksjonalitet den representerer. Tar vi en titt på den gule merkelappen leser vi at ikonknappen står for ” Check Browser Compatibility”, altså sjekk nettleserkompatibilitet. Men fortsatt gjenstår spørsmål i forhold til hva kompatibiliteten eventuelt sjekkes mot. I brukerguiden finner jeg ut at at knappen lar meg sjekke om mine CSS er kompatible på tvers av ulike nettlesere. Vi ser altså at ikonknappen er semantisk motivert, men i forhold til det funksjonelle innholdet er uttrykket svært fattig. Sentralt innhold som ”CSS” og ”nettleser” kommer ikke frem, og kanskje henger det sammen med egenskapene til disse sememene. En sentral handling i funksjonen er ”sjekke kompatibilitet”, men gitt problemer med å finne egnede semantiske markører til projisering i ”kompatibilitet” er det kun ”sjekke” som har funnet sin uttrykksform. Vi ser altså et gjentagende problem angående innhold med egenskaper som ikke enkelt lar seg projisere ikonisk, noe som betyr stor avstand til det ”intuitive grensesnittet” slik jeg har definert det i forhold til forståelse i denne oppgaven.

Ikonknapp nr. 9 er plassert som i stor grad semantisk motivert, men allikevel nokså uforståelig. Leser vi den gule merkelappen tilknyttet uttrykksinstanser står det ”View Options”. Til tross for denne opplysningen er det uklart hva bildet i ikonknappen skal forestille. En titt i brukerguiden avslører at knappen lar meg sette alternativer for ”Code view” og ”Design view”, inkludert hvilket visningsvindu som skal være over

det andre. Gitt denne opplysningen er det vanskelig å se at innholdet ikonet står for (funksjonen) og innholdet den projiserer semantiske markører fra har noe til felles. Dette likner situasjonen vi så i ikon nr. 6 angående bildet av øyet; altså at de projiserte egenskapene hentet fra andre sememer enn noen av de som kan sies å utgjøre handlingen som representeres. Dette gjør det selvfølgelig vanskelig for forståelsen, og fjerner uttrykksinstansene et stykke vekk fra intuitivitetsidealet. I dette tilfellet (nr. 9) er det vanskelig å bestemme hvilket innhold ikonet faktisk henter motivasjon fra, jeg synes det ser ut som en stilisert punktliste. I likhet med ikon nr. 6 (og flere andre) kan antakelig problemet tilbakeskrives til de abstrakte fenomenene ikonet skal representere, særlig tatt i betraktning at det er et menyhode. ”Å sette alternativer for Code view og Design view” utelukker projisering av semantiske markører fra faktiske eksempler på visningsvalg da dette kan misforstås som det eneste visningsalternativet.

Ikonknapp nr. 10 med bilde av en trekant og en tag, har en gul merkelapp med skriften ”Validate markup” hektet på seg når markøren føres over den. Men denne tilleggsinformasjonen er ikke nok for å kunne si noe om den semantiske motivasjonen. I brukerguiden står det om ikonknappen at den lar meg validere det aktuelle dokumentet eller en valgt tag. Tagen er altså motivert av ”markup/tag”, mens den grønne trekanten antakelig skal stå for funksjonen ”å validere”. Vi har sett tagens toposensitive visuelle egenskaper har blitt projisert som del av uttrykksinstans før (kontroll 4 og 5) for å representere kode, men det er vanskeligere å definere motivasjonen bak den grønne ”play”-knappen. Nettopp det at det likner en fremoverpekende pil gjør at jeg skjønner det dreier seg om slags bevegelse eller operasjon, men i forhold til intuitivitetsidealet er det flere sememer som må ”gi seg selv” for at jeg skal forstå funksjonaliteten. Problemet er nok en gang å finne passende egenskaper ved en abstrakt operasjon, ”å validere”, som lar seg projisere i et ikon og som kan gi grunnlag for forståelse. I likhet med konseptet ”CSS” er operasjonen ”å validere” en gåte uten kunnskap om koding og kodestandarder. Vi ser derfor at forståelse krever kunnskap jeg som novise ikke er i besittelse av, og som jeg kun kan få gjennom tilleggsdokumentasjonen.

Ikonknapp nr. 11 med bilde av de to pilene har merkelappen ”File management”. Brukerguiden forteller at det dreier seg om funksjonaliteten å sjekke filer ”inn og ut”, og med denne opplysningen er det noe enklere å forstå den semantiske motivasjonen

bak pilene. De går i hver sin vei. Men, det interessante med denne motivasjonen er at den er hentet fra en språkfigur ”sjekke inn og sjekke ut”, som i seg selv kan kalles en metafor. Vi ser altså at den semantiske motivasjonen ikke er hentet fra handlingen selv, men måten handlingen uttrykkes på gjennom skrift og tale. I forhold til intuitivitetsidealet lønner det seg antakelig ikke å uttrykke språkfigurer ved hjelp av ikoner når skriftspråket selv er mer presist. ”Sjekke inn og sjekke ut” er ikke konvensjonelt kodet til to piler i hver sin retning. Ellers finner vi at det sentrale sememet ”fil” ikke blir forsøkt uttrykt i ikonet, det på tross av at det er et allment konsept på tvers av applikasjoner. Kanskje er dette igjen et utslag av at ikonet er et menyhode og derfor må være abstrakt; ofte finner vi at filikoner ikke uttrykker det generelle konseptet ”fil”, men er proxyikon for en spesiell fil eller filtype. Igjen ser vi at konseptene ”forvaltning” (management) og ”fil” mangler spesielle (konvensjonaliserte) semantiske markører som kan projiseres og gjøre ikonene ”intuitive”, og at en mellomløsning er valgt i forhold til konseptet ”forvaltning” ved at semantiske egenskaper ved innholdet til en språkmetafor er projisert. Som alle operasjoner (eksekvering av programkode) involverer filforvaltning toposensitive egenskaper som tid og forandring som egentlig kun kan uttrykkes dynamisk. Men ofte kan det være nok å projisere semantiske markører ved subjektene som er involvert for å gi antydning angående operasjonen (hvis markørene er konvensjonelt knyttet til en gitt funksjon), men som vi ser i forbindelse med filforvaltning er ”fil” i seg selv et abstrakt konsept uten distinkte semantiske markører som lar seg projisere, og uten en klar funksjon knyttet til seg.

6.7.4 Tilbakemelding og kommunikasjon

Et viktig poeng i forhold til forståelse av det grafiske brukergrensesnittet er hvordan tilbakemelding og kommunikasjon kan hjelpe til i den semiotiske prosessen. Jeg har allerede aktivt brukt de gule merkelappene som et hjelpemiddel i analysen på x-aksen. I forhold til uttrykksinstansene 1,2,3,4 og 5 ser vi at visuelle toposensitive egenskaper ved funksjonaliteten kommer til syne som visuell forandring i dokumentvinduet, eller som i inputfeltet (nr. 2) gjennom synlige bokstaver. Ved en eksplorativ ”prøv og feil”-taktikk er det dermed mulig å forstå funksjonaliteten i disse uttrykksinstansene. På liknende måte gir uttrykksinstansene 6,7,8,9,10 og 11 tilleggsinformasjon som kan brukes i den semiotiske prosessen gjennom de menyene som dukker opp ved interaksjon. Denne informasjonen uttrykkes gjennom en meny av funksjoner som alle deler visse egenskaper som menyhodet forsøker å formidle. Slik kan jeg ved ekstrakoding kanskje komme frem til både innholdet til menyhodet, samt forstå funksjonaliteten i det enkelte menyvalg. Denne gradvise utvidelsen av kodekunnskapen basert på stadig ny tilleggsinformasjon vil jeg ikke analysere i detalj, men nevne fordi det både er en måte å ta hånd om kompleksitet på samt sikre brukerkontroll.

6.7.5 Oppsummering

Oppsummerer vi graden av forståelse i Dokumentverktøylinjen finner vi, ikke overraskende, at tegnfunksjoner som brukes på tvers av applikasjoner (og som dermed står for et allment innhold) er enklest å forstå. Dette gjelder også de segmenterte kontrollene der språket er ferdigkodet i forhold til et allment innhold og dermed forankrer den ikoniske uttrykksinstansen (som ikke er konvensjonelt kodet) til et presist innhold. Et kjennetegn ved ikonene som er vanskeligere å forstå er at de i egenskap av å være menyhoder representerer et abstrakt innhold som ikke har noen konvensjonelt kodet ikonisk representasjon. Vi ser derfor at i den grad de semantiske markørene som er projisert gjenkjennes, knyttes de til et innhold som kun er løst forbundet med den funksjonen uttrykksinstansen faktisk står for (i følge brukerguiden).

6.8 Panel

Rundt Dokumentvinduet finner vi tre vinduer av typen panel. Panel er en generell term brukt av Apple for å beskrive tilgjengelige hjelpevinduer som inneholder viktige kontroller og valgmuligheter med effekt på det aktive dokumentet eller et markert område. Ulike typer paneler kan ha mer spesifikke navn, som Dreamweavers Innsettingslinje og Egenskapsinspektør.

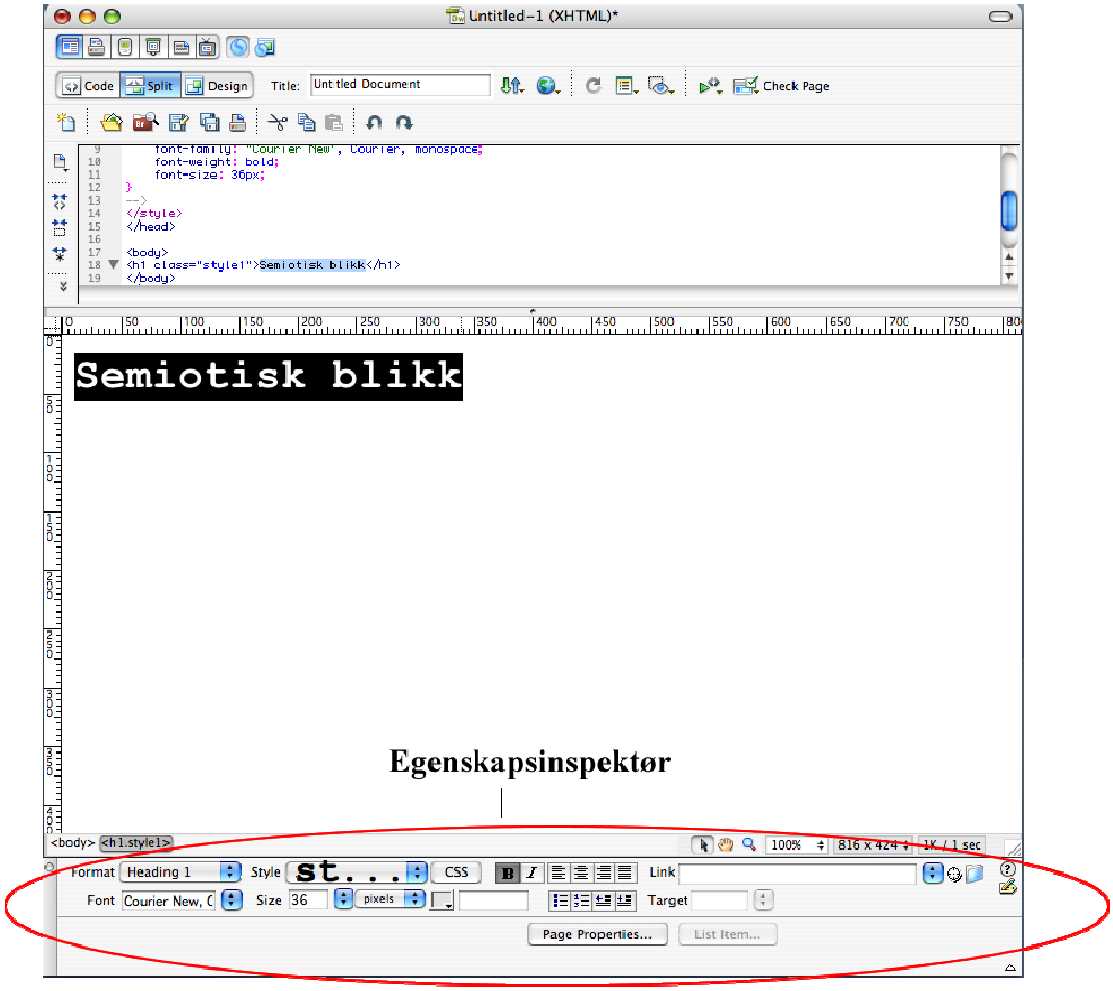
Men fordi paneler tar opp skjermplass bør de ikke brukes hvis gode alternativer finnes. Brukeren kan åpne flere paneler samtidig og de flyter over og ved siden av dokumentvinduet. Panelene blir brakt til front når applikasjonen er aktiv, men forsvinner når den er inaktiv.

”Inspector Windows”, i Dreamweaver kalt ”Property inspector” (Egenskapsinspektøren, se Figur 10), reduserer rotet i grensesnittet ved å plassere tilleggsinformasjon og innstillinger i et separat vindu som kan skjules eller vises etter brukerens preferanser. Muligheten for brukeren til å ha preferanser angående hva som vises på skjermen og til en viss grad hvordan applikasjonen oppfører seg, reduserer kompleksiteten. Ved å tilby preferansevalg tillates både noviser og eksperter å forme applikasjonen til å passe egne behov.

I det følgende avsnittet blir Egenskapsinspektøren i Dreamweaver beskrevet nærmere med henblikk på strukturelle og visuelle karaktertrekk.

6.9 Egenskapsinspektøren (property inspector)

Egenskapsinspektøren (property inspector) er en strukturerende uttrykkstype som viser de mest vanlige egenskapene til et valgt objekt i Dokumentvinduet og gir muligheter til å redigere disse. Uttrykksinstansene i Egenskapsinspektøren varierer avhengig av det valgte objektet. Under vises Egenskapsinspektøren når et tekstobjekt er valgt i Dreamweaver.



Figur 10 med bilde av Egenskapsinspektøren

I midten av vinduet ser vi en horisontal separator som deler flaten i to. Over linjen ligger de fleste uttrykksinstansene. Øverst til venstre ser vi to standard pop-up vinduer med introduksjonsetikett (introductory label) med teksten ”Format” og ”Style”. Det trykkbare området er fargekodet med blå bakgrunn, og ved interaksjon dukker en rekke menyvalg opp. Under ”Format” og ”Style” ser vi to lignende pop-up menyer med introduksjonsetikett ”Font” og ”Size”, men disse har standard inputfelt og en liten avstand til den blå trykkbare firkanten. Vi ser altså at små visuelle hint i form av romlig avstand samt skyggelegging og skarpe kanter markerer forskjellig interaksjonsform og muligheter. I midten ser vi en firkantet CSS-knapp, en liten rektangulær kontroll kalt fargekilde/brønn (color well), og ved siden av denne et inputfelt for tekst. I midten finner vi også ti stiliserte rektangulære ikoner med definerte rammer. Disse ligner standard ikoner for å manipulere tekst slik jeg kjenner de fra andre applikasjoner, som Microsoft Word. Stilen og romlig plassering tett inntil hverandre gjør at jeg straks vet hvilke som er relatert til hverandre. Til høyre ser vi to

dominerende inputfelt med hver sin introduksjonsetikett og pop-up meny. Også her brukes blå fargekoding for å tiltrekke oppmerksomheten mot området som skal trykkes på for å få opp menyen. Helt til høyre finnes fire bitte små ikoner med særdeles ulik form, noe som gir inntrykk av at de er rasket sammen og plassert på den eneste ledige plassen. Ser vi nøye etter er avstanden mellom mappeikonet og spørsmålstegnikonet større enn avstandene ellers, og dette indikerer at mappen og "Point to File"-ikonet (det runde), er relatert til pop-up menyen. Men denne strukturelle relasjonen er ikke konvensjonell for mappeikonet. Spørsmålstegnet og ikonet under er relatert romlig, men andre visuelle kjennetegn gir ingen hint om hvorfor. Om de er relatert innholdsmessig er tema for andre del av analysen.

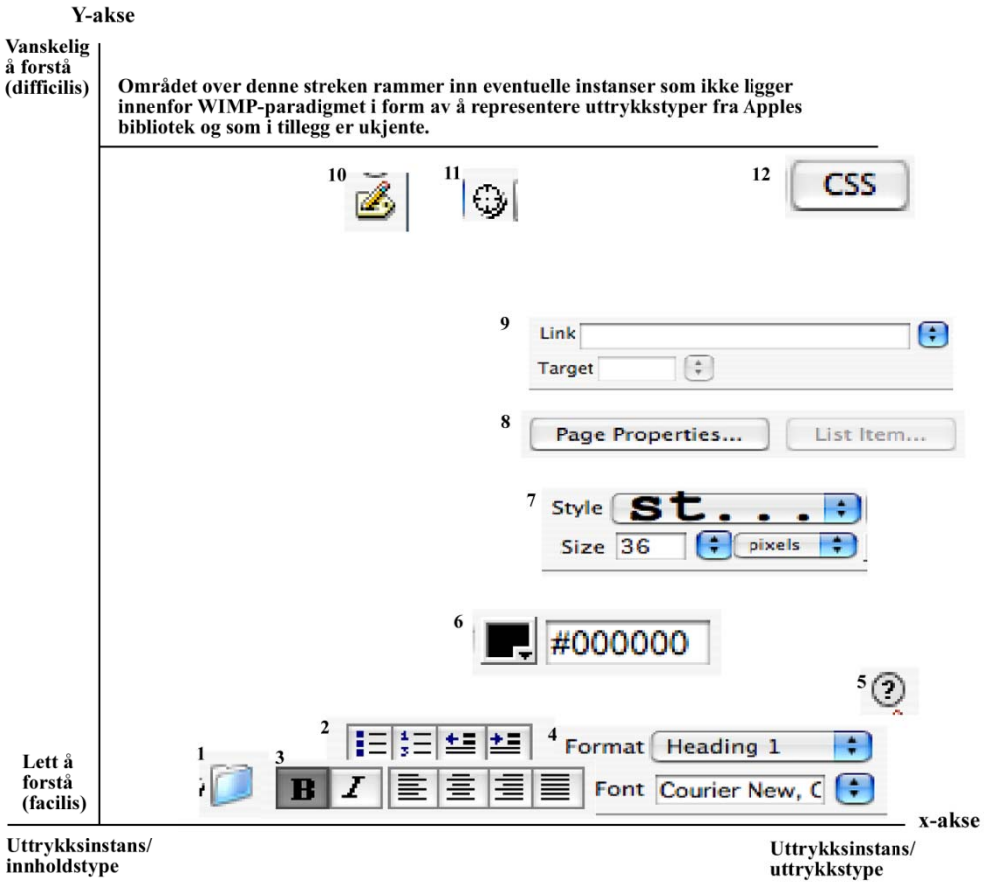
Under den horisontale separatoren er det to firkantede knapper, hvor av den ene er aktiv og den andre skyggelagt og dermed ikke valgbar. Plasseringen av de to knappene gir inntrykk av at de er relatert også på innholdsplanet. Nederst til høyre er en liten trekant som vender oppover og som dermed ser noe forskjellig ut fra avsløringstrekantene i f.eks. Filpanelet. Men funksjonelt likner den ved at den lukker den nedre delen av vinduet.

Summerer vi den strukturelle og visuelle beskrivelsen ser vi nok en gang at uttrykksinstansene kan kategoriseres etter hvilke av Apples definerte uttrykkselementer de tilhører, de er dermed del av WIMP-paradigmets karakteristiske uttrykksformer. Et interessant trekk er at mange av de definerte uttrykkselementene ikke har krav på seg til å være semantisk motivert, tvert om er de fleste tilfeldig koblet til sitt innhold ved hjelp av tekst. I tillegg ser vi at den romlige plasseringen og fargekoding gir essensiell informasjon i forhold til funksjon og interaksjonsform. I neste avsnitt skal jeg undersøke i hvilken grad jeg forstår uttrykksinstansene i Egenskapsinspektøren. Dette vil kunne si noe om Dreamweavers forhold til Apples ideal om det intuitive brukergrensesnittet.

6.9.1 Aksesystemet

Under vises aksesystemet der uttrykksinstansene i Egenskapsinspektøren er kartlagt. Vi ser at mange av instansene er ganske store slik at det er vanskelig å plassere dem nøyaktig på aksene, men forhåpentligvis vil teksten bringe klarhet i hvor de hører hjemme. Vi minnes at aksesystemet totalt sett kan si noe om forholdet mellom

uttrykksform og forståelse i forhold til en viss type innhold. Denne gangen i forhold til de egenskapene tekstobjektene er i besittelse av.



Figur 11. Aksesystemet med Egenskapsinspektøren viser grad av forståelse og forholdet mellom uttrykk og innhold.

6.9.2 Grad av forståelse

Tar vi en titt på den totale ansamlingen av uttrykksinstanser i aksesystemet, ser vi at forholdsvis mange er plassert som lette å forstå. Dette gjelder mappeikonet (nr. 1), de ti stiliserte rektangulære ikonene (nr. 2 og 3), samt det standard pop-up vinduet med introduksjonsetikettene ”Format” og ”Font” (nr. 4). Felles for dem alle er at de er standard uttrykksformer som går igjen på tvers av applikasjoner, samt at de er enkle å forstå hver for seg. Mappen er en velkjent død metafor hentet fra skrivebordsmiljøet Mac OS implementerer. De rektangulære ikonene visualiserer hvordan resultatet av interaksjonen vil se ut, mens pop-up menyene i nr. 4 bruker allmenne ord som ”Format” og ”Font” som lettfattelig introduksjon til verdiene som vises. I tillegg kommer at inputfeltet i ”Font” faktisk viser den aktive fonten. Spørsmålstegnet (nr. 5) har erobret sin plassering med bakgrunn i gjeldene konvensjon om at spørsmålstegn

indikerer hjelp. Fargebrønnen er noe vanskeligere grunnet den forvirringen som kan oppstå på grunn av visningen av heksadesimalene, men siden den er så konvensjonell er det fortsatt rimelig enkelt å forstå. Det kan virke underlig at pop-up menyene i nr 7 er plassert som nokså vanskelig, men dette henger sammen med usikkerheten rundt hva ”Style” egentlig står for og om ”Size” eventuelt definerer stilen. Men vi ser at menyen faktisk viser den aktive stilen, så det er lagt opp til at jeg skal forstå hvilke egenskaper den har. Den firkantede knappen med teksten ”Page Properties” samt den skyggelagte knappen med teksten ”List Item” er plassert som noe vanskelige fordi teksten er så generell at det er umulig å vite hva jeg får. Det store inputfeltet for tekst, samt det lille i nr. 9 er vanskelige å forstå fordi ”Link” og ”Target” ikke er allmenne ord og jeg derfor ikke vet hvilke verdier som skal inn i inputfeltet. Felles for de firkantede knappene og inputfeltene (nr. 8 og 9) er at den vesentligste informasjonen angående deres funksjon avsløres av ord som enten forstås eller ikke. Det betyr at det er lite rom for abduksjon i det øyeblikket jeg ikke skjønner hva skriften sikter til. Ikonene nr. 10 og 11 er plassert som svært vanskelige å forstå fordi det er vanskelig å se hva de faktisk forestiller, samt å komme på funksjonalitet som passer i forhold til hvor de er plassert i vinduet. CSS-knappen er avhengig av at jeg forstår ordet ”CSS”, et spesialisert akronym som ikke kan kalles allment. Dessuten er knappen plassert såpass ensomt at det er vanskelig å beslutte om den står i sammenheng med de nære omgivelsene.

6.9.3 Forholdet mellom uttrykk og innhold

Vi husker at uttrykksinstansene i Egenskapsinspektøren tilhører uttrykkstyper definert av Apple og at de dermed er del av WIMP-paradigmets karakteristiske uttrykksformer. I denne delen av analysen vil jeg ta sikte på å kunne si noe om hvordan Dreamweavers funksjonalitet formidles gjennom disse uttrykksformene ved å karakterisere korrelasjonen mellom uttrykksinstansen og innholdet.

Et kjapt blikk på aksesystemet forteller at de fleste uttrykksinstansene i Egenskapsinspektøren er plassert ganske langt til høyre på aksen. Begynner vi med motivasjonen bak ikon nr. 1, mappeikonet, finner vi at denne er plassert som sterkt motivert av innholdstypen fordi den nettopp projiserer karakteristiske semantiske markører hos en mappe, som dens firkantede form og øreklaff for navngiving. Den er nokså stilistisk gjengitt, så den burde antakelig vært dyttet litt til høyre hadde det vært

plass. Jeg er vant til at filer finnes i mapper, så jeg antar den er plassert der for at jeg skal kikke etter filer (selv om jeg ikke kan bestemme hva filen eventuelt skal brukes til). Tar vi en titt på den gule merkelappen som hører med, står det nettopp ”Browse for File”. En titt i brukerguiden avslører at det er vanskelig å finne opplysninger angående denne mappen, men gjennom en simpel test (valg av en tilfeldig fil) ser jeg at filen jeg velger bli plassert i inputfeltet til ”Link” og at den markerte teksten blir forandret til en lenke. Jeg har altså valgt en fil den markerte teksten lenker til, men dette er essensiell informasjon angående funksjonaliteten ikonet i seg selv ikke avslører. Ikonet er altså lett å forstå isolert sett (se etter filer), men det er vanskeligere å skjønne at det står i sammenheng med uttrykksinstans nr.9 (”Link” inputfelt) fordi det ligger andre uttrykksinstanser i mellom, og fordi det er ukonvensjonelt plassert strukturelt sett.

Tar vi for oss de ti stiliserte rektangulære ikonene (nr. 2 og 3) er disse plassert som delvis motivert av innholdstypen fordi de projiserer visuelle semantiske markører ved resultatet av den operasjonen de står for. Vi ser at ikonene egner seg godt til å formidle disse visuelle toposensitive egenskapene ved innholdstypene (ulike former for visuell fremtoning for skrift).

Tar vi for oss den standard pop-up menyen med introduksjonsetikettene ”Format” og ”Font” (nr. 4), er det vanskelig å oppdage noen projiserte semantiske markører bortsett fra ved innholdet i inputfeltet til ”Font”, som avspeiler visuelle egenskaper ved den fonten som er valgt i øyeblikket. Dette er et interessant eksempel på at også uttrykksinstanser som står for eksplisitte handlinger kan omgjøres til objekter slik at de visuelle egenskapene i uttrykkstypen anses som projiserte semantiske markører. Uttrykksinstans er både tilfeldig og ikke tilfeldig knyttet til sitt innhold avhengig av hvilken kontekst det fortolkes i. Ved ”klassisk” interpretasjon er uttrykksinstansen tilfeldig koblet til sitt innhold gjennom navnet på fonten, ved fortolkning i verktøymodus er uttrykksinstansen omgjort til uttrykkstypen tekst, og projiserer da essensielle egenskaper ved uttrykkstypen selv. Men gitt at skrift gjerne fortolkes på ”klassisk” vis (jeg antar at det er tilfeldig koblet til innholdet), ser vi at funksjonaliteten til disse to pop-up vinduene er ganske åpenbar takket være etikettene, mye fordi et godt allment språklig vokabular er arvet fra typologien for dette

semantiske feltet. I tillegg kommer selvfølgelig at de samme termene brukes på tvers av applikasjoner.

Ikon nr. 5, det runde ikonet med et spørsmålstegn, er en veldefinert uttrykkstype i WIMP-paradigmet, men tilfeldig koblet til sitt innhold. Det er derfor interessant at det kalles et ikon uten å være ”ikonisk” (d.v.s. semantisk motivert). Usikkerheten angående dens funksjon er knyttet til den romlige plasseringen som gjør det vanskelig å vite nøyaktig hva man får hjelp til. Ved en test finner jeg ut at det dukker opp et vindu med generell hjelp angående hvordan man manipulerer egenskapene til tekst gjennom Egenskapsinspektøren, og mer presist hvordan man kan velge og praktisere enten HTML-formattering eller CSS-formattering. Spørsmålstegnet står dermed for et innhold som ikke uten videre kan knyttes til de andre instansene.

Fargebrønnen (nr. 6), det vil si den svarte firkanten, er plassert som delvis semantisk motivert fordi den projiserer en vesentlig egenskap ved sememet som visualiseres i øyeblikket, nemlig den gitte fargen. Den lille pilen nederst er som nevnt før en konvensjonell måte å vise at det finnes en pop-up meny ved interaksjon, men den er tilfeldig koblet til dette innholdet og derfor instans av en uttrykkstype. Heksadesimalene er likeså tilfeldig koblet til den fargen de representerer, og til sammen er derfor uttrykksinstans nr. 6 plassert midt på x-aksen.

Vi ser at pop-up menyene i nr. 7, med introduksjonsetikettene ”Style” og ”Size”, for det meste er tilfeldig koblet til sitt innhold og at instansene derfor utgår fra en uttrykkstype. Den svake dragningen mot venstre skyldes at menyen til ”Style” viser visuelle egenskaper ved den stilen som er valgt i øyeblikket. Dette er det samme som vi så for ”Font” i uttrykksinstans nr. 4. Etter litt prøving og feiling (fordi det ikke er enkelt å finne frem i brukerguiden angående Egenskapsinspektøren) skjønner jeg at ”Style” blant annet defineres av valgene i ”Size” og ”Font”, og at den definerte stilen kan velges for annen tekst ved en annen anledning. Det er altså snakk om å definere CSS-style, om enn med kun noen få attributter. Vi ser dermed at begrepet ”Style” og det konkrete eksempelet på en stil i form av et markert objekt, gjør det mulig å forstå noe av konseptet bak CSS. Men siden begrepet ”CSS” ikke brukes, er jeg ikke klar over dette!

De firkantede knappene med teksten ”Page Properties...” og ”List Item...” utgår begge fra en uttrykkstype og er derfor tilfeldig koblet til sitt innhold. Teksten på begge knappene er allmenne ord slik at jeg forstår hva det betyr, men innholdet er så abstrakt og minimalt at jeg ikke kan vite nøyaktig hva som vil dukke opp. Det interessante er jo at jeg heller ikke finner annet vokabular enn det som er brukt på knappene for å beskrive innholdet i dialogboksen, det er rett og slett en rekke generelle egenskaper ved den aktuelle siden og muligheter til å manipulere disse. Knappen ”List Item...” forutsetter valg av en liste for å være brukbar. Til tross for et svært diffust språk på knappen gjør kravet om at en liste må være markert sitt til at jeg kan avgrense hypotesene angående hva knappen kan brukes til. Det dreier seg om ulike manipulerbare egenskaper ved listen som er valgt.

De to inputfeltene med introduksjonsetikettene ”Link” og ”Target” (nr.9) utgår begge fra en uttrykkstype og er tilfeldig koblet til sitt innhold. De er derfor plassert helt til høyre på x-aksen (størrelsen på uttrykksinstansen gjør at dette kommer litt dårlig frem i aksesystemet). Selv om jeg vet hva begrepene ”Link” og ”Target” betyr er det vanskelig å forstå nøyaktig hva som forventes som input i inputfeltene. Hvis ”Link” krever nettopp en adresse, hva er da ”Target”? Brukerguiden forteller at ”Target” definerer hvor en lenket side skal åpnes (samme vindu, nytt vindu eller spesiell ramme i et vindu). Denne funksjonen forutsetter derfor kunnskap om ”framesets”, et konsept som ikke er standard for en novise.

Øverst til venstre finner vi ikonet med et bilde av noe som ligner en blyant og en merkelapp (nr. 10). Tilhørende dette ikonet er den gule merkelappen ” Quick Tag Editor”, og det er først ved forankring gjennom skriftspråket jeg forstår hva ikonet representerer. Blyanten står for ”editor”, et lite hurtigprogram for å redigere ”tags”, representert ved hjelp av den typen tag (merkelapp) man har på kofferten. Plutselig blir altså den døde metaforen, tag, avslørt som metafor! Siden sememet ”tag” forstått i HTML-kontekst konvensjonelt uttrykkes ved hjelp av språkmetaforen ”tag” eller ved hjelp av projiserte toposensitive visuelle egenskaper som ”<>” (som ikke er metafor), er det ingen etablert korrelasjon mellom bildet av en koffert-tag og ”<>” uten at den gitte språkmetaforen ”tag” er til stede. For at jeg skal forstå koffert-tagen må jeg derfor selv bidra med ordet ”tag” i interpretasjonsprosessen, noe som krever at jeg forventer å finne en metafor for sememet ”tag” (noe jeg ikke gjør fordi jeg er vant til

at sememet ”tag” uttrykkes ikonisk ved hjelp av ”<>”). Alternativt må ordet ”tag” bli gitt på andre måter, og vi så at løsningen ble den gule merkelappen der ordet er skrevet (og som skapte ”aha”-opplevelsen). Tar vi for oss blyanten ser vi at også denne er en metafor der jeg ut fra verktøymodus antar at essensielle egenskaper ved blyantens virkemåte skal gi hint om funksjonen til ”Quick Tag Editor”. ”Å skrive” er en opplagt handling tilknyttet blyanten, ”å redigere” en mindre opplagt undergren av skrivingen. Nok en gang er problemet å finne essensielle egenskaper som lar seg projisere og som er konvensjonelt knyttet til handling (redigere).

Ikon nr. 11, kalt ”target icon” i brukerguiden, har knyttet til seg en gul merkelapp ”Point To File”. Ved litt testing finner jeg ut at jeg kan dra en pil fra denne til den filen jeg vil lenke til. Gitt opplysningene om at dette er et ”target icon” skjønner jeg at den runde formen med en prikk i midten skal forestille en måltavle. Essensielle egenskaper ved innholdet til ”måltavle” (en metafor) kan altså overføres og smeltes med handlingen ”Point To File”, og jeg antar det siktes til handlingen ”å skyte på blink”. Å ”peke til fil” og ”å skyte på blink” har til felles et bevegelse mot et gitt mål. Problemet med den stiliserte metaforen er at egenskapene som er projisert ikke er distinkte nok, noe som henger sammen med at det ikke eksisterer noen konvensjonell kode for hvordan en måltavle skal vises ikonisk. I det handlingen utføres kommer det til syne en blå pil som veldig direkte visualiserer essensielle toposensitive egenskaper ved handlingen ”Point To File”, nemlig bevegelse mot et mål. Denne underforståtte handlingen gir dermed i likhet med direkte manipulasjon følelsen av en intuitiv forståelse av det som foregår på skjermen.

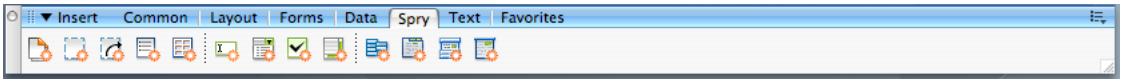
Øverst til høyre finner vi knappen med teksten ”CSS”. Den er tilfeldig koblet til sitt funksjonelle innhold, og som ved tekst generelt er det gjerne et enten/eller-spørsmål i forhold til forståelse så lenge det ikke finnes noe grunnlag for ekstrakoding i form av kontekstuelle eller strukturelle forhold. Den gule merkelappen beskriver funksjonaliteten med ordene ”Open CSS Panel”, og det er akkurat det knappen gjør, åpner CSS-panelet.

6.9.4 Oppsummering

Forsøker vi å oppsummere graden av forståelse i sammenheng med hvilke uttrykksformer Egenskapsinspektøren bruker for å formidle egenskaper ved objektene, ser vi at mange av tegnfunksjonene er lette å forstå fordi de står for funksjonalitet som er vanlig i tilknytning til tekstobjekter på tvers av applikasjoner. Denne funksjonaliteten er knyttet til et allment nedarvet og vel utviklet skriftspråk som brukes som uttrykksform. Men forståelsen avtar etter hvert som språket blir mer domenespesifikt, nettopp fordi spesialisert funksjonalitet knyttes til et spesialisert språk. Men vi ser samtidig at heller ikke en ikonisk representasjon sikrer forståelsen, og dette henger i vesentlig grad sammen med at innholdet som skal uttrykkes er domenespesifikt og ikke konvensjonelt knyttet til en ikonisk uttrykksform. I tillegg er det vanskelig å komme med gode hypoteser angående innholdet fordi funksjonaliteten som skal uttrykkes mangler opplagte semantiske markører som lar seg projisere.

6.10 Innsettingslinjen

Under vises et bildeutsnitt av Innsettingslinjen (Insert bar i Figur 3) i Dreamweaver der taben ”Spry” er åpen. Innsettingslinjen er i standardinnstilling plassert over Dokumentvinduet, men under menylinjen. Den kan flyttes rundt, så plasseringen er ikke essensiell for forståelsen.



Figur 12. Innsettingslinjen med Sprykategorien åpen.

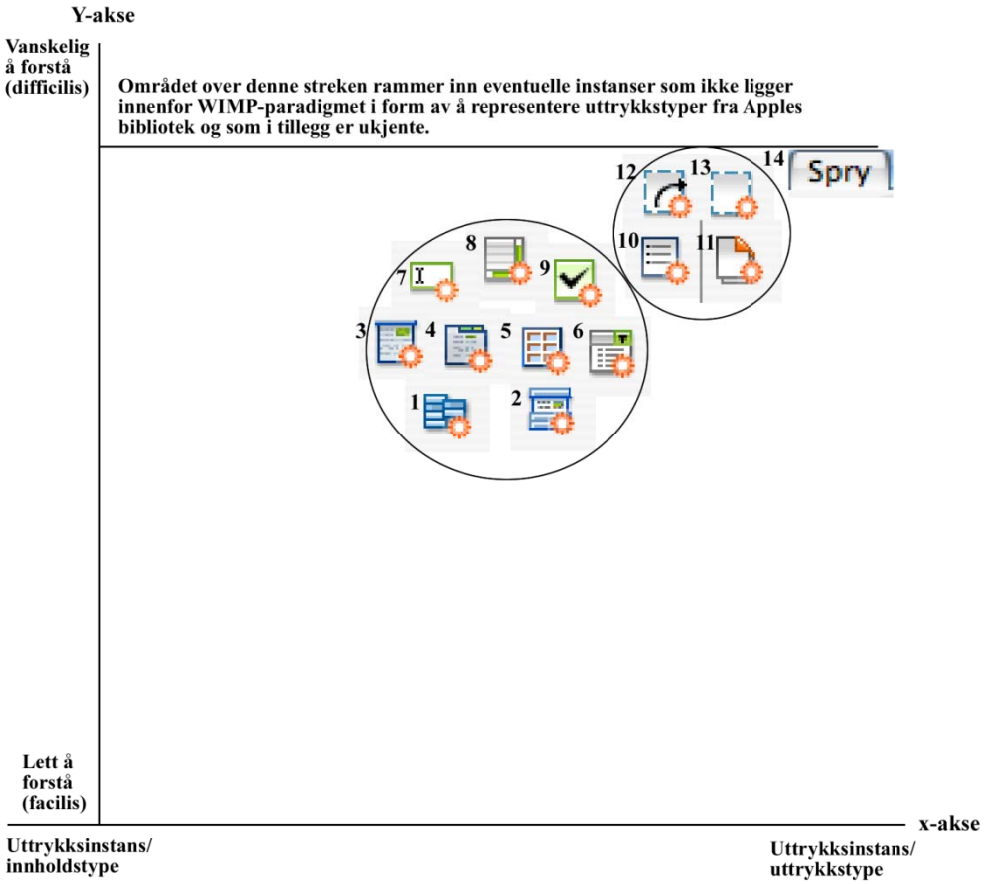
Innsettingslinjen har i likhet med andre strukturerende uttrykksinstanser rektangulær form, og rommer ikonknapper for å lage og sette inn objekter, som for eksempel tabeller, bilder og andre applikasjonselementer. Ikonknappene er organisert i flere kategorier som kan aktiveres og vises ved å trykke på tabene på toppen av linjen. Vi ser at det aktive kategorivalget vises ved hjelp av fargekoding, der taben har samme farge som omgivelsene til de knappene den organiserer. Både knappene og tabene er skilt ved hjelp av diskrete vertikale separatorer. Tabenes plassering langs samme horisontale akse, den delte bakgrunnsfargen samt enighet om tekst som uttrykksform på tabenes kategorier forklarer og definerer relasjonene dem imellom. Tabenes

strukturerende funksjon defineres i relasjon til knappene, som gjennom fargekoding og inkluderende ramme (taben er inkludert), knyttes til kategorien og hverandre. Sammenligner vi Innsettingslinjen med de segmenterte kontrollene i Dokumentverktøylinjen ser vi at fargekodingen ikke er konsistent i det blåfargen indikerer ”ikke valgt” i Innsettingslinjen og ”valgt” i Dokumentverktøylinjen.

Gjennomgangen av Innsettingslinjens strukturelle og visuelle karaktertrekk viser at den følger Apples standard og implementerer uttrykkstypene verktøylinje og ikonknapper. Dette er et premiss for å kunne si noe om hvordan Dreamweavers funksjonalitet formidles gjennom WIMP-paradigmets uttrykksformer (i dette tilfellet ikonknapper), samt i hvilken grad uttrykksinstansene er forståelige.

6.10.1 Aksesystemet

Under vises aksesystemet som visualiserer i hvilken grad jeg forstår uttrykksinstansene, samt forståelsens sammenheng med måten funksjonaliteten formidles på gjennom tilfeldig eller motivert korrelasjon til innholdet. Spry-kategorien representerer en viss type innhold uttrykt gjennom ikonknapper. Hvor godt dette innholdet passer til en ikonisk uttrykksform er som vanlig del av diskusjonen rundt graden av overlapp mellom de projiserte semantiske markørene og det innholdet som faktisk skal formidles. På neste side vises aksesystemet:



Figur 13. Aksesystemet med Spryikonene i Innsettingslinjen kartlagt for å illustrere grad av forståelse samt forholdet mellom uttrykk og innhold.

6.10.2 Grad av forståelse

En titt på aksesystemet avslører at alle ikonknappene er vanskelige å forstå for meg som novise. Sirklene er tegnet inn for å markere at ikon-knappene egentlig har samme plassering. Uttrykksinstansene i den store sirkelen er plassert som litt mer forståelige enn de i den lille sirkelen fordi det er mulig å dra kjensel på noen av de stiliserte projiserte semantiske markørene her og knytte dem til uttrykkstyper jeg har sett i grensesnitt før. Men mangelen på forståelse er allikevel vesentlig knyttet til det faktum at jeg ikke har sett uttrykksinstansene før, samt at jeg aner at de røde rundingene i nederste høyre hjørne markerer en spesiell samhørighet rundt et konsept jeg ikke har greie på. Uttrykksinstansene i den lille sirkelen gir enda færre holdepunkter for vågale hypotetiske gjetninger, og den samme usikkerheten melder seg angående de røde rundingene. Dette inntrykket av å mangle kunnskap forsterkes av taboverskriften ”Spry”, som ikke gir noen holdepunkter for å avklare de felles egenskapene hos tegnfunksjonene som inngår i kategorien. Ser vi på de enkelte

tegnfunksjonene er det ut fra disse heller ikke mulig å komme til noen entydig konklusjon angående hva ”Spry” egentlig betyr. Ordet er i seg selv tilfeldig koblet til sitt funksjonelle innhold i verktøymodus, og blir enten forstått eller ikke. Derfor er det henvist til en plass øverst i høyre hjørne. Den gjensidige sammenligningen av tegnfunksjonene skaper altså en forsterket usikkerhet idet hverken skriftspråket eller ikonene gjenkjennes og kan kobles til et sannsynlig innhold. Det indikerer at innholdet hverken kan oppsummeres i et enkelt allment ord, eller kan uttrykkes billedlig gjennom ikoner. Neste avsnitt om x-aksen analyserer forholdet mellom uttrykk og innhold nettopp med tanke på å finne hvilke problemer som knytter seg til forholdet mellom den gitte uttrykksformen (ikoner organisert under en kort overskrift) og det gitte innholdet.

6.10.3 Forholdet mellom uttrykk og innhold

Diskusjonen i forbindelse med x-aksen dreier seg om forholdet mellom innholdet slik det forstås gjennom uttrykksinstansen (hvis jeg forstår noe), og det funksjonelle innholdet som faktisk skal formidles. Siden overskriften for alle ikonknappene er ”Spry”, er en forståelse av dette begrepet avgjørende for å skjønne resten av uttrykksinstansene i Innsettingslinjen. Begrepet er tilfeldig knyttet til sitt innhold, så jeg må gå til eksterne kilder for å finne ut hva det betyr. Kunnskapsforlagets Engelsk-Norsk ordbok definerer ordet i ”klassisk” kontekst slik: ”aktiv, rask, kvikk, våken” (Kunnskapsf. Engelsk, 2007). Gitt at ordet ikke er tilfeldig valgt, er det derfor rimelig å anta at alle tegnfunksjonene i Sprykategorien er knyttet til disse egenskapene på en eller annen måte. Ser vi i Dreamweavers brukerguide etter en mer passende beskrivelse basert på den definerte verktøymodusen, går det fram at ”Spry” er et rammeverk bygd opp av et Javascript og CSS bibliotek. Elementene i biblioteket er interaktive og kan brukes for å vise dynamisk XML-data som oppfriskes lokalt (det vil si at hele siden ikke trenger og oppfriskes hvis kun ett spry-element skal oppdateres). Denne grunnleggende forklaringen viser at en forståelse av konseptet ”Spry” krever forhåndskunnskap angående en rekke begreper, som Javascript, CSS og XML. Basert på denne kunnskapen er det så mulig å sette de enkelte uttrykksinstansene inn i sin rette sammenheng. Mangelen på en overhengende analogi som kan knytte de nye begrepene til kjent kunnskap og dermed sette ting i sammenheng blir åpenbar når den domenespesifikke funksjonaliteten skal uttrykkes gjennom et særdeles sparsommelig uttrykksregister. I gjennomgangen av hver enkelt

uttrykksinstans skal vi se nøyere på hvilke semantiske markører som projiseres fra hvilke innholdstype(r) for så å relatere denne innholdstypen til det innholdet (funksjonaliteten) som faktisk skal formidles (slik det kommer fram av Dreamweavers brukerguide).

For å slippe å gjenta meg selv vil jeg samle de uttrykksinstansene som ligner på hverandre med hensyn til hvilken type innhold de henter motivasjonen fra og hvilken type funksjon de representerer.

De første uttrykksinstansene jeg vil analysere er ikonknappene nr. 1,2,3,4 og 5. Disse projiserer toposensitive visuelle semantiske markører fra en innholdstype som generelt kan karakteriseres som elementer for innsetting. Ikon nr. 1 har merkelappen ”Spry Menu Bar” og skal ligne nettopp en stilisert menylinje med et sett navigerbare menyknapper som viser submenyer når en bruker fører markøren over en av knappene. Ikon nr. 2 ”Spry Accordion” skal på lignende måte vise et sett sammenleggbare paneler som åpnes og lukkes ved å klikke på taben til panelet. Ikon nr. 3 ”Spry Collapsible panel” viser kun ett sammenleggbart panel, mens ikon nr. 4 ” Spry Tabbed Panels” viser et sett paneler organisert i klikkbare taber. Det siste ikonet i gruppen, ” Spry Table” (nr. 5), er et av de enkleste å kjenne igjen og viser nettopp sentrale visuelle egenskaper ved en tabell.

Et interessant fenomen ved beskrivelsen av alle disse uttrykksinstansene er bruken av metaforer. Ordene ”menu”, ”accordion” og ”panel” er alle metaforer som spiller på essensielle toposensitive visuelle egenskaper ved det opprinnelige innholdet for å forklare utseende til de grafiske elementene de betegner. Ikonene derimot er egnet til å vise nettopp disse visuelle egenskapene og trenger dermed ikke gå veien om metaforer. Det vil si, vi har sett at det ikke er enkelt å kjenne igjen og tolke de projiserte semantiske markørene i ikonene, men dette henger sannsynligvis sammen med at de ikke er del av en konvensjonell kode idet det mangler regler for hvordan denne type innhold skal vises ikonisk. Kanskje er det også her en mangel på distinkte visuelle egenskaper som kan skille ett ikon fra et annet og peke utvetydig til ett bestemt semem. Men i kombinasjon er ikonet og den tekstlige metaforiske beskrivelsen gjensidig forankrende.

Denne gjennomgangen av de fem første ikonene kunne tyde på at de slett ikke burde plasseres som så vanskelige å forstå, og at innholdet er godt egnet for en ikonisk framstilling. Det dreier seg rett og slett om knapper til å sette inn elementene som er avbildet. Men som nevnt tidligere markerer den røde rundingen og ordet ”Spry” at det er noe ekstra ved disse elementene; noe ”aktivt, raskt, kvikt og våkent” for å sitere ordboken. Elementene er resirkulerbare ”widgets” som tillater interaksjon, som at en meny åpner en submeny, et panel legges sammen, eller at en tab åpnes ved et klikk. Disse toposensitive egenskapene kommer ikke frem gjennom ikonet, nettopp fordi de er dynamiske og kun kan vises i tid. Allikevel er ikke dette en avgjørende mangel; vi ser nemlig at de elementene (eller uttrykkstypene) som er visualisert i ikonene er del av Apples bibliotek over standard uttrykkstyper, og måten de fungerer på er dermed en del av etablert paradigmekompetanse (WIMP). Det å vite hva ”Spry” står for er dermed ikke avgjørende for å kunne bruke disse funksjonene, men mangelen på kompetanse viser seg å bli problematisk i det det innsatte elementet skal manipuleres.

Vi ser at alle uttrykksinstansene i den store sirkelen skjuler implementasjonen av delvis kompleks funksjonalitet ved elementene (scripts og styles) brukeren strengt tatt ikke trenger ha kunnskap om for å kunne sette dem inn i Dokumentvinduet. Men det er én uttrykksinstans i den store sirkelen som krever tilleggskunnskap også for en simpel innsettingsoperasjon, nemlig ”Spry Table”. En vanlig tabell er gjerne ikke særlig dynamisk, det er derfor ikke gitt gjennom kjennskap til Apples standard uttrykkstyper hvilke dynamiske egenskaper en Sprytabell har. Det finnes to typer Sprytabeller, en enkel og en mastertabell som binder til seg en ”detail region”. Det interessante er den siste typen som henter inn data dynamisk og oppdateres ved interaksjon (mastertabell med detaljregion). For å få til dette er brukeren nødt til å foreta noen initierende handlinger, som å definere ”spry region” og velge Spry dataset. Disse handlingene krever grunnleggende kompetanse som ikke enkelt uttrykkes gjennom et ikon og som heller ikke formidles gjennom en tilgjengelig analogi. Dreamweaver forsøker å kompensere for manglende uttrykksmuligheter ved hjelp av grupperinger skilt av vertikale separatorer. Tabellikonet er sortert sammen med ikonknappene vi finner i den lille sirkelen, men det er ingen uttrykk som formidler hva som er spesielt med disse og som dermed utgjør grupperingsparameteret.

Tar vi for oss de neste fire ikonknappene, nr. 6,7,8 og 9, ligner de mistenkelig på de fem første ved å projiserer toposensitive visuelle semantiske markører fra innsettingselementer. Men i tillegg implementerer disse ikonknappene en valideringsfunksjon som sjekker innholdet brukeren velger eller legger inn. Denne er i utgangpunktet ikke en naturlig konnotasjon til de projiserte uttrykkstypene og må derfor uttrykkes for og oppdages. Det skal sies at Dreamweaver gjennom strukturell organisering og fargekoding (grønn) markerer at disse fire ikonknappene er forskjellig fra resten og har en felles egenskap. Men hva det er kommer ikke frem.

Valideringsfunksjonen handler om å sjekke om brukerens valg eller inputdata er gyldig i forhold til noen valgte parametre. Det vil altså si at ikonknappene ikke kun krever en enkel innsettingsoperasjon, men også trenger definerte verdier å forholde seg til som brukeren må legge inn. Poenget er ikke om operasjonen er lett eller vanskelig, men at det ikke gis noen hint i ikonknappen selv om at denne funksjonaliteten er del av elementet som representeres.

Når det gjelder ikonknappene i den minste sirkelen er det vanskeligere å si nøyaktig hvor motivasjonen bak de projiserte semantiske markørene er hentet fra. Tar vi for oss funksjonaliteten til ikonknappene for å finne svar, ser vi at de har gule merkelapper med navnene ”Spry Repeat List” (nr. 10), ”Spry XML Data Set” (nr. 11), ”Spry Repeat” (nr. 12) og ”Spry Region” (nr. 13). Språket viser seg dermed ikke å være særlig behjelpelig med å indikere hva som er felles for de fire ikonknappene eller forklare hva funksjonaliteten er. Ikon nr. 10 ligner riktignok på en liste når jeg ser ordet ”List” nevnt, men etter mitt skjønn er ikke lister særlig dynamiske, så det aner meg at det ligger funksjonalitet skjult jeg ikke får tak på. Ved hjelp av Dreamweavers brukerguide forstår jeg det slik at disse fire ikonknappene (samt Spry table) kan brukes til å vise data ved hjelp av ”Spry”. Men knappene kan ikke trykkes på i tilfeldig rekkefølge; først må jeg identifisere dataene jeg skal bruke ved å trykke på ”Spry XML Data Set”, så må jeg trykke på ”Spry Region” for å definere en region der eventuelt ”Repeat List”, ”Spry Repeat” eller ”Spry table” kan få plass. Og ser vi på plasseringen av ikonknappene i Innsettingsbaren er de plassert i denne rekkefølgen (gitt vestlig leseretning). Men ikonknapper er normalt ikke ment å leses sekvensielt og dekkes dermed ikke av min etablerte strukturelle kodekompetanse (WIMP-paradigmet).

Vi ser at ingen av ikonknappene projiserer egenskaper det er enkelt å knytte til den funksjonaliteten de representerer. Ikon nr. 10 kan ligne en liste, men den vesentligste funksjonen i form av dynamisk tilfang av data for visning i ulike former for lister kommer ikke frem. Ikon nr. 11 kan ligne et ark, men det er vanskelig å knytte dette til den initierende funksjonen ikonknappen har i forhold til alle de andre ikonknappene den er gruppert sammen med. I tillegg er det lite ved arket som kan knyttes til en XML datakilde. Ikon nr. 12 ser ut til å ville markere ”Repeat” ved hjelp av en pil, samt at region er en blå stiplet ramme. Det faktum at jeg kan tolke ikonknappen i den retningen tilsier at de projiserte semantiske markørene til dels kan settes i sammenheng med innholdet og at de derfor ikke kan kalles helt tilfeldig. Men det er ingen ting som tilsier at jeg skulle kunne skjønne at knappen representerer en mulighet til å formattere regioner (datastrukturer) for å vise dynamisk data. Ser vi denne ikonknappen i sammenheng med nr. 13 finner vi at prinsippet om kontinuitet er etterfulgt ved at region nok en gang er fremstilt som en blå stiplet ramme. Men stikkordet ”Region” er ikke presist nok til å kunne knyttes til en spesifikk handling, i dette tilfellet definisjonen av en type region som omslag (wrapping) for dataobjekter (objekter som viser XML data).

6.10.4 Oppsummering

Et påtakelig trekk ved analysen av Sprykategorien i Innsettingslinjen har vært misforholdet mellom den nødvendige grunnleggende kompetansen (kodekjennskap) jeg som bruker må ha for å benytte funksjonaliteten, og min faktiske kunnskap. Dette misforholdet går rett i kjernen av dilemmaet angående pedagogisk veiledning og profesjonell bruk. Ideelt sett skal begge hensyn kunne integreres i det grafiske brukergrensesnittet, men analysen viser at det ikke gjøres noen forsøk på å formidle denne grunnleggende kompetansen gjennom Dreamweavers grensesnitt. Ikonknappene fungerer mer som gjenkjennelige snarveier enn informative uttrykk angående bruksområdet. Den vesentligste kunnskapen er ikke uttrykt i det hele tatt (Sprykonseptet), og funksjonaliteten har ikke opplagte semantiske markører som enkelt lar seg projisere. Inkludert i den grunnleggende kunnskapen er også en logisk arbeidsflyt. Vi så at flere av ikonknappene er ment å leses sekvensielt fordi funksjonaliteten som uttrykkes må komme i en viss logisk rekkefølge. Men denne måten å lese ikoner på er ikke standard i WIMP-paradigmet, så problemet med å

visualisere sekvens av handlinger i selve grensesnittet er påtakelig. Denne mangelen på forståelse av sammenhenger og arbeidsflyt kan knyttes til mangelen på overhengende analogi/metafor (designprinsipp 3.3.1) eller annen form konseptuell ramme.

7. KONKLUSJON

Med utgangspunkt i en undring angående hvordan datamaskinens komplekse funksjonalitet formidles, har vi sett at det eksisterer et sett retningslinjer og idealer for utforming av grafiske brukergrensesnitt som Dreamweaver i stor grad følger for å nærme seg intuitivitetsidealet. Tar vi for oss forskningsspørsmål 1 angående hvordan Dreamweaver gjør seg nytte av Apples designprinsipper for å nærme seg intuitivitetsidealet, har vi sett at disse sørger for at jeg som bruker gjenkjenner den overhengende strukturen i grensesnittet, samt de forskjellige uttrykkstypene med definerte visuelle egenskaper og tilknyttet interaksjonsform. Vi kan dermed si at WIMP-paradigmet slik det kommer til uttrykk gjennom Apples retningslinjer bidrar med en nødvendig kompetanse på veien mot det ”intuitive” grensesnittet.

Men til tross for at Dreamweaver ser ut til å implementere Apples generelle retningslinjer, ser vi at det i flere tilfeller skorter på forståelsen i møte med mange av uttrykksinstansene. Noe av svaret på hvorfor dette er tilfellet er å finne i analysen av forholdet mellom uttrykk og innhold, samt de strukturelle kjennetegnene.

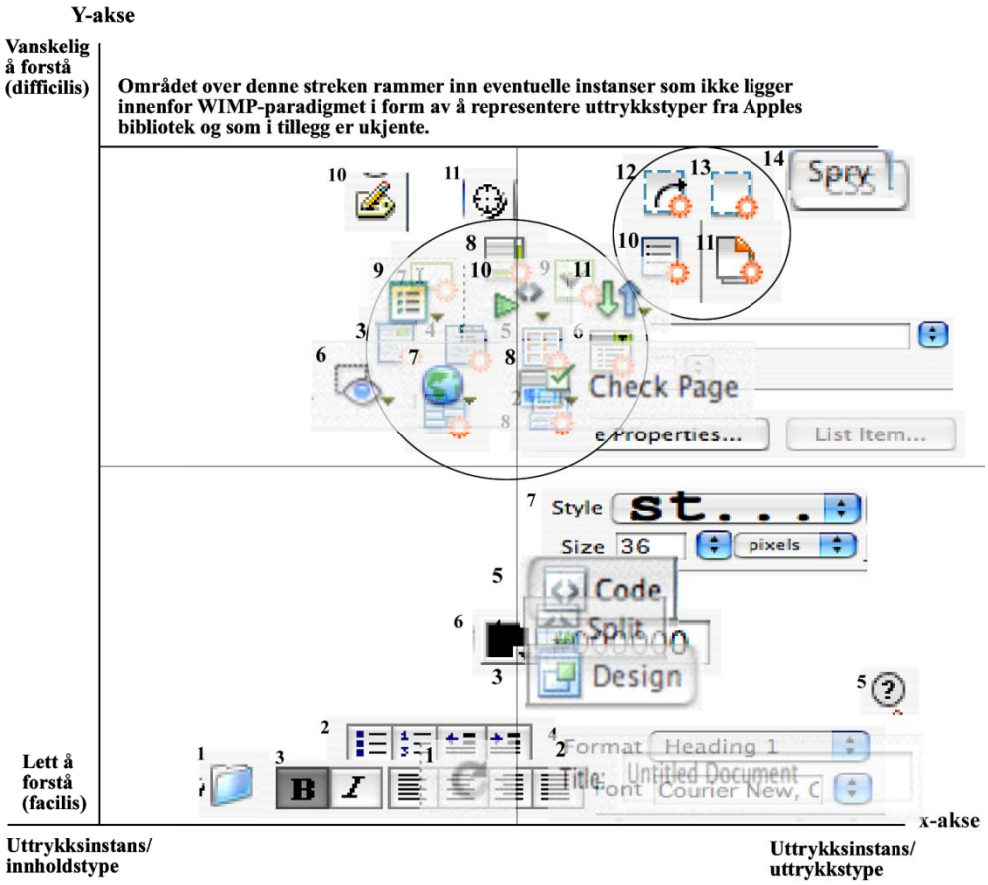
I det følgende vil jeg oppsummere noen av funnene angående matchen mellom Apples foreskrevne uttrykksformer og Dreamweavers funksjonalitet og bruksmuligheter. Disse funnene besvarer forskningsspørsmål 2 angående hva forholdet mellom Dreamweavers uttrykksform og egenskaper ved det funksjonelle innholdet sier om grunnlaget for forståelse. Jeg vil først oppsummere hva som danner grunnlag for forståelse, for så å se på årsaker til manglende forståelse.

7.1 Grunnlag for forståelse gjennom ikke domenespesifikk funksjonalitet

Første del av analysen dreide seg om hvordan WIMP-paradigmet sikrer forståelsen av de underforståtte handlingene tilknyttet de egenproduserte objektene. Dette er handlinger som går på tvers av applikasjoner, noe som vil si at funksjonaliteten ikke er domenespesifikk. Vi kom frem til at uttrykkstypen i seg selv var det denotative innholdet når objektet ble fortolket i verktøymodus, og at både eksplisitte og underforståtte handlinger ble knyttet til objektet basert på erfaring.

Premisset om kjennskap til WIMP-paradigmet og dermed underforståtte handlinger, samt den direkte tilbakemeldingen der toposensitive egenskaper som bevegelse, tid og retning visualiseres fra applikasjonens side i det jeg utfører handlingen, sikrer at underforståtte interaksjonsformen oppleves som ”intuitiv”. Men samtidig er det klart at det er en begrenset mengde underforstått funksjonalitet som kan være generell nok til å kunne deles blant mange applikasjoner i tillegg til å kunne visualiseres i tid og rom, og dermed bli en del av paradigmet.

Mens de underforståtte handlingene er innlærte interaksjonsformer som følger med WIMP-paradigmet, er de eksplisitte handlingene både paradigmetro, allmenne handlinger, og ny domenespesifikk funksjonalitet med krav om spesiell uttrykksform. Ser vi på aksesystemet under, der alle uttrykksinstansene er inkludert, finner vi at også forståelse av de eksplisitte handlingene er knyttet til etablert kodekompetanse gjennom ikke domenespesifikk funksjonalitet og gjenkjennelige uttrykksinstanser.



Figur 14. Aksesystemet med alle de analyserte uttrykksinstansene representert.

Tar vi for oss uttrykksinstansene under den vannrette skillelinjen midt i aksesystemet, ser vi at mange av funksjonene har et nokså allment innhold som har en standardisert uttrykksform jeg kjenner til etter å ha erfart ulike applikasjoner. Mange av uttrykksinstansene er hentet fra Egenskapsinspektøren, og det er ikke tilfeldig at egenskapene til et skriftobjekt (med lange historiske røtter), er sterkt kodet i skriftspråket og en del av det konvensjonelle ideenes leksikon (se kap. 4.5). I tillegg til de ikke domenespesifikke egenskapene ved tekstobjektene, ser vi at annen funksjonalitet representert ved mappeikonet, fargebrønnen og tittel-linjen er gjenkjennelig og delt mellom flere applikasjoner. Vi kan derfor si at den ikke domenespesifikke funksjonaliteten med konvensjonelt tilknyttet uttrykksform er enkel å forstå fordi den er en del av det etablerte WIMP-paradigmet.

7.1.1 Forståelse gjennom figurale og vektorielle egenskaper

I tillegg til at en del av tegnfunksjonene er allmenne, ser vi også at typisk toposensitive figurale og vektorielle egenskaper egner seg til ikoner fordi de er lettere å projisere og kjenne igjen enn andre egenskaper. Vi ser at visuelle kjennetegn ved uttrykkstypen tekst og arrangering av tekst, er uttrykt gjennom velegnede uttrykkstyper som stiliserte ikoner og menyer. Fargebrønnen er et ypperlig eksempel på hvordan visuelle kjennetegn lar seg projisere, mens de tre segmenterte kontrollene for visning av dokumentet er eksempel på det motsatte der det er teksten som forankrer innholdet sammen med den innbyrdes plasseringen. ”Design” og ”Code” er for abstrakte konsepter for ikonisk fremstilling, og de projiserte semantiske markørene er derfor heller stiliserte eksempler (instanser) av uttrykkstypen som er for upresise til å stå alene. Et motsatt tilfelle er begrepet ”Style” som er for upresist til å stå alene, men som forankres ved at projiserte visuelle kjennetegn ved nettopp stilen projiseres ved siden av.

7.1.2 Forståelse gjennom relasjoner og konsekvenser

Et felles kjennetegn ved de forståelige menyene og kontrollene er at de gir rom for intern sammenligning ved at funksjonaliteten dreier seg rundt variasjoner over et forståelig tema eller objekt. Menyene forankres gjennom den språklige merkelappen (Style, Size, Format, Form), mens de stiliserte kontrollene forankres ved at temaet (tekst) har gjenkjennelige projiserte egenskaper. Dette medfører en betydelig innsnevring av mulig innhold til de ulike valgene. I tillegg har erfaringene med ett valg (og den umiddelbare konsekvensen for det valgte objektet) overføringsverdi til de andre valgene innen samme strukturerende uttrykkstype. Kjenner jeg til størrelsen på ”Size 36” er det mulig å gjette resultatet av de andre valgene.

7.1.3 Sammenheng mellom forståelse og uttrykkstype?

Relaterer vi de to aksene i forhold til uttrykksinstansene som er kategorisert som greie å forstå, er det vanskelig å få øye på en klar sammenheng. Vi ser at allerede kjennskap til tegnfunksjonen er det essensielle, uansett motivasjon. Men det er en tendens til at mange av uttrykksinstansene havner på høyre side av x-aksen. Gitt det begrensede tilfanget av uttrykksinstanser i analysen er det umulig å si noe generelt om sammenhengen, men jeg våger meg på en antakelse om at siden skriftspråket med en konvensjonelt tilfeldig kobling til innholdet er så vel kodet i vår kultur, og siden

gjenkjenning av ikoner med semantisk motivasjon er begrenset til en viss type innhold (med mindre de også er del av vårt felles semantiske univers), er det sannsynligvis lettest å være presis gjennom skriftspråket så lenge dette er konvensjonelt tilknyttet det aktuelle innholdet. Vi skal i neste avsnitt se at problemene oppstår når det er ukonvensjonelt innhold som skal uttrykkes på liten plass. Hvilke uttrykkstyper velges i det ny kode skal lages? Hvorfor forstår jeg ikke den ukonvensjonelle koden?

7.2 Domenespesifikt innhold og mangel på forståelse

Tar vi en titt over den horisontale skillelinjen står de fleste uttrykksinstansene for funksjonalitet som må kunne kalles domenespesifikk. Den manglende forståelsen kan dermed knyttes til at det er vanskelig å finne allmenne ord for spesialisert funksjonalitet, eller at det er vanskelig å finne egenskaper ved innholdet som lar seg projisere der WIMP-paradigmet foreskriver ikoner. I det følgende vil jeg ta for meg hvilke egenskaper ved det domenespesifikke innholdet som gjør det vanskelig å uttrykke gjennom de etablerte uttrykkstypene.

7.2.1 Innhold med dynamiske egenskaper

Funksjonalitet dreier seg gjerne om en handling, og handling involverer forandring, og forandring er en egenskap som er vanskelig å uttrykke statisk. Tatt i betraktning at handlingen ”insert” er gitt i Innsettingslinjen, ligger forholdene til rette for en ikonisk uttrykksform for et konkret innhold. Og riktig nok lykkes jeg delvis i å gjenkjenne de projiserte figurale egenskapene (store sirkelen), men problemet melder seg i forbindelse med den røde sirkelen. Denne står for de toposensitive dynamiske egenskapene involvert i Sprykonseptet, og den faktiske dynamiske funksjonaliteten som hører til hvert enkelt innsettingselement lar seg ikke uttrykke statisk. Et annet eksempel er ikonene i Dokumentverktøylinjen som projiserer egenskaper fra et innhold som ikke direkte kan knyttes til den handlingen de står for. Dette gjelder for eksempel at jordkloden står for handlingen ”å forhåndsvise”, pilene for ”å forvalte” og trekanten for ”å validere”. I Egenskapsinspektøren så vi målskiven som skulle stå for ”å peke”.

7.2.2 Abstrakt innhold uten egenskaper som lar seg projisere

Problemet med å uttrykke dynamiske egenskaper statisk knytter seg ofte også til det faktum at handlingen involverer abstrakte fenomen med egenskaper som ikke lar seg projisere. Dette er tilfelle med ikonene i Dokumentverktøylinjen, som alle har et abstrakt innhold. Dette siste poenget er en konsekvens av at ikonene fungerer som menyhoder og derfor er nødt til å være generelle nok til å inkludere alle menyvalgene de er knyttet til. ”Nettleser” og ”Fil” er for eksempel knyttet til en for generell innholdstype (overbegrep i ordboken) til å kunne vises ikonisk. Det samme gjelder begrepet ”Spry” som dekker et såpass stort, komplekst, abstrakt semantisk felt at det ikke kan forklares ved hjelp av noen få projiserte semantiske markører i et enkelt ikon.

7.2.3 Abstrakt innhold uten konvensjonelt tilknyttede ord

Begrepet ”Spry” minner oss om et dobbelt problem. Nemlig at konseptet det uttrykker hverken har et konvensjonelt språklig uttrykk eller har semantiske markører som lar seg projisere. Vi ser øverst i høyre hjørne flere eksempler på dette. Akronymet ”CSS” er hverken allment eller enkelt å uttrykke ikonisk, og ikonene vi ser i den lille sirkelen har verken knyttet til seg noe lett forståelig vokabular eller gjenkjennelige semantiske markører som lar seg knytte til innholdet.

7.2.4 Definert sekvens av funksjoner

Et siste poeng angående innholdet i Sprykategorien gjelder hvordan deler av funksjonaliteten er kausalt knyttet sammen. Vi ser dermed eksempel på et trippelt problem i den minste sirkelen, der innholdet er både dynamisk, abstrakt og inngår i en definert sekvens av funksjonalitet. Som vi så i analysen fortolkes ikonene som selvstendige uttrykksinstanser der posisjon på skjermen mer antyder slektskap enn en gitt syntaktisk rolle ulik de andres. De er derfor ikke egnet til å uttrykke en nødvendig sekvens av handlinger. Dette er et interessant eksempel på at Dreamweaver går ut over WIMP-paradigmets gitte regler ved å omdefinere uttrykkstypenes rolle i grensesnittet.

7.2.5 Mangel på overhengende konseptuell modell

Problemet med å forstå sekvens av handlinger henger nøye sammen med mangelen på en overhengende konseptuell modell av arbeidsprosessene som er involvert i arbeidet

med websites. Vi så i analysen av det overhengende stuktureringsprinsippet i Dreamweaver at prinsippet ”reflekter brukerens mentale modell” ikke var fulgt, og at en viktig grunn til dette var at brukere gjerne ikke har en ferdig etablert mental modell av Dreamweavers domene. Dette henger i vesentlig grad sammen med egenskaper ved Dreamweavers domene. Problemet med å forstå uttrykksinstansene blir dermed dobbelt vanskelig i det de både er ugjenkjennelige og ikke tilhører en etablert konseptuell modell. Mer presist består dilemmaet i at jeg på den ene siden må etablere en noenlunde realistisk mulighetshorisont (basert på erfaring og kunnskap) for å kunne gjette funksjonaliteten, og på den andre siden må uttrykksinstansene være såpass knyttet til det funksjonelle innholdet at det er mulig å trekke forbindelsen. Det vil si at jeg optimalt sett burde kunne bedrive overkoding ved hjelp av eksisterende kodekjennskap (konseptuell modell) og underkoding ut fra uttrykksinstansens utseende. Hvis begge strategier feiler er det ingen grobunn for abduktive slutninger.

7.3 Med ordet som anker

Et interessant fenomen er de ikonene som forholdsvis enkelt lar seg forklare med ord. Dette gjelder for eksempel de fleste ikonene i Dokumentverktøylinjen, men også for eksempel ”Quick Tag Editor” og ”Point to File” i Egenskapsinspektøren. Vi ser dermed at språket gjerne er mer presist enn ikonene når det gjelder å forankre innholdet. Det er ikke sjelden en ”aha”-opplevelse har kommet i det den gule merkelappen dukker opp. Men hvorfor brukes da ikke skriftspråket mer? Dette henger sammen med grensesnittets doble funksjon: det skal både fungere som kontrollpanel og samtidig kommunisere funksjonaliteten i applikasjonen. I det de språklige de kjente språklige virkemidlene som bruk av allerede etablert kunnskap gjennom analogier og metaforer ikke lenger er lett tilgjengelige, er paradigmets grunnlag borte. Dette fordi disse språklige virkemidlene gjør plassøkonomisering mulig slik at det ”intuitive” kontrollpanelet realiseres. Lange forklaringer øker nok forståelsen, men er grenseløst irriterende for en utdannet bruker som kun trenger å gjenkjenne uttrykksinstansen: igjen og igjen og igjen og igjen.

7.4 Vilkår for forståelse versus vilkår for effektivitet

Vi har sett en interessant løsning på dilemmaet mellom profesjonell bruk og pedagogisk veiledning gjennom de gule merkelappene. Disse dukker kun opp etter noen sekunder, slik at profesjonelle brukere kan unngå dem. I tillegg så vi at velkomstvinduet tok sikte på å sile ut og advare nybegynnerne. Analysen har vært med på å bekrefte at Dreamweaver mer fungerer som et kontrollpanel enn en pedagogisk inngangsportal til webdesign og at dette henger nøye sammen med det domenespesifikke innholdet.

7.5 Vilkår for forståelse versus allmenn tilgjengelighet

Tatt i betraktning vilkårene for forståelse, er det et interessant spørsmål som reises angående vilkårene for adgang til ny funksjonalitet. I den grad overgangsanalogienes tid er over, vil det kreves mer opplæring for å ta i bruk nye applikasjoner. Ny funksjonalitet krever nytt språk, samtidig som nettopp nytt språk fostrer ny funksjonalitet. Risikoen er derfor at avansert funksjonalitet nok en gang blir forbeholdt en spesialisert elite. Har vi i mente at den første Macintoshen ble solgt ved hjelp av argumenter som henspilte på demokrati og lav brukerterskel, er dette litt av et paradoks.

7.6 Oppsummering av hovedproblemstilling

La oss ta opp igjen hovedproblemstillingen: Hvordan fungerer Apples retningslinjer for design som rammeverk for å kommunisere Dreamweavers funksjonalitet og bruksmuligheter på en forståelig måte? Vi er nå i stand til å oppsummere dette spørsmålet ved å peke på at Apples retningslinjer fungerer som et godt rammeverk i forhold til forståelse når applikasjonens domene lar seg forme etter en sammenhengende konseptuell modell, og når funksjonaliteten som skal formidles kan uttrykkes presist (er sterkt kode) gjennom WIMP-paradigmets uttrykksformer på liten plass. Jeg har over pekt på hvilke egenskaper innholdet må være i besittelse av for at dette skal være mulig.

8. AVSLUTTENDE REFLEKSJONER

I dette avsluttende kapitlet vil jeg kort peke på et interessant trekk i utviklingen av grafiske brukergrensesnitt, samt si noe om analyseapparatet jeg har brukt.

8.1 Post OS?

Antakelig er det utviklingstrekk som peker i en mer åpen retning også. Et interessant fenomen er den friheten som åpner seg i forhold til grensesnittdesign i forbindelse med en stadig mer avansert funksjonalitet i nettlesere. Kanskje har operativsystemet utspilt sin rolle som normsetter innen grensesnittdesign på skjermen til den personlige datamaskinen?[18] I det nettleseren blir den foretrukne "front end" applikasjonen i et server-klient system, er det ikke lenger avgjørende hvilket operativsystem som brukes. I så måte er det nettleseren og signifikasjonssystemet som gjør seg gjeldende her som definerer hvordan designen skal se ut. Men spørsmålet som stilles i denne oppgaven er fortsatt relevant; vil uttrykksformene i en standard nettleser kunne utrykke ethvert domene på en forståelig måte?

8.2 Ecos semiotiske teori og grafiske brukergrensesnitt

Gjennom analysen av Dreamweavers uttrykksformer har Ecos semiotiske teori vist seg å være et nyttig apparat for å kunne peke på noen av forutsetningene for at forståelse skal finne sted. Jeg vil påstå den egner seg godt til grafiske brukergrensesnitt, fordi disse fremviser en rekke ulike uttrykksformer med en presis intensjon bak seg fra designerens side. Det begrensede universet med godt definert innhold er dermed en forutsetning for å gjøre Ecos produksjonsteori om til en interpretasjonsteori, mens det komplekse signifikasjonssystemet gjør at analyseapparatet gir interessante resultater.

[18] I forhold til for eksempel mobiltelefoner ser det ut til at operativsystem fortsatt definerer grensesnittstandarden. Det gjelder ikke minst Apples iPhone.

Referanser:

Andersen, P.B. (1990): A Theory of Computer Semiotics. Semiotic approaches to construction and assessment of computer systems. Cambridge: Cambridge University Press

Adobe Systems Inc. (2007): "*Adobe Dreamweaver CS3. User guide."* URL: http://livedocs.adobe.com/en_US/Dreamweaver/9.0/index.html [pdf-fil lastet ned 06.05.2009]

Adobe to acquire Macromedia. URL: http://www.adobe.com/aboutadobe/invrelations/adobeandmacromedia.html [Lesedato 06.05.2009]

Apple Computer Inc. (1987): *Human Interface Guidelines: The Apple Desktop Interface.* Addison-Wesley Publishing Company

Apple Computer Inc. (2008): *Apple Human Interface Guidelines*. URL: http://developer.apple.com/documentation/UserExperience/Conceptual/AppleHIGuidelines/XHIGIntro/XHIGIntro.html [pdf-fil lastet ned 06.05.2009]

Bokmålsordboka: "brukergrensesnitt". Oslo: Universitetsforlaget, 2006. URL: http://www.dokpro.uio.no/ordboksoek.html [Lesedato 06.05.2009]

Campbell-Kelly, Martin og William Aspray (2004): *Computer: a history of the information machine.* 2.utgave, The Perseus Books Group

Chandler, Daniel (2002): *Semiotics – the basics.* London: Routledge

Eco, Umberto (1979): *A theory of semiotics*. Bloomington: Indiana University Press.

Eco, Umberto (1984): *Semiotics and the philosophy of language.* London: Macmillan

Fagerjord, Anders (2003): *Rhetorical convergence: earlier media influence on web media form.* Avhandling (dr. art.), Universitetet i Oslo

Grudin, Jonathan (1990): *The Computer reaches out: The historical continuity of interface design.* CHI ´90 Proceedings. Association for Computing Machinery (ACM)

Horton, William (1994): *the icon book. Visual symbols for computer systems and documentation.* John Wiley & Sons

Jørgensen, Anker Helms (2004):"Fra beregning til æstetik: computeren i det akademiske landskab set i et HCI-perspektiv" I: *Digitale verdener.* Ida Engholm og Lisbeth Klastrup (red.). København: Gyldendal

Kluge, Anders (2005): *Progressive interaction design for the metamedium: An investigation into interactive meaning-making.* Avhandling (dr. scient.), Universitetet i Oslo

Krippendorff, Klaus (2006): *the semantic turn – a new foundation for design*. Florida: CRC Press

Kunnskapsforlagets blå fremmedordbok: ”interaksjon” og ”objekt”. Oslo: H. Aschehoug Co. (W. Nygaard) og Gyldendal, 2004.
URL: http://www.ordnett.no/ordbok.html [Lesedato 06.05.2009]

Kunnskapsforlagets Engelsk blå ordbok*:* ”spry”. Oslo: H. Aschehoug Co. (W. Nygaard) og Gyldendal, 2007.
URL: http://www.ordnett.no/ordbok.html [Lesedato 06.05.2009]

Manovich, Lev (2001): *The language of new media.* Cambridge: MIT Press

Norman, Donald (1988): *The Design of Everyday Things.* New York: Basic Books

Pew, Richard W. (2008): ”Evolution of Human-Computer Interaction: from Memex to bluetooth and beyond”. I: *The Human-Computer Interaction Handbook. Fundamentals, Evolving Technologies and Emerging Applications.* Julie A. Jacko og Andrew Sears (red.). New York: Lawrence Erlbaum

Preece, Jennifer, Yvonne Rogers, Helen Sharp (2002): *Interaction design: beyond human-computer interaction.* John Wiley & Sons, Inc

Raskin, Jef (2000): The Humane Interface- New Directions for Designing Interactive Systems. Addison Wesley

Store norske leksikon: ”paradigme”. URL: http://www.snl.no/paradigme/filosofi [Lesedato 06.05.2009] Opphavsmann: Lars Fredrik Händler Svendsen.

Thurk, Jessica og Gary Alan Fine (2003): “The Problem of Tools – Technology and the Sharing of Knowledge”. I: Acta Sociologica, Jun 2003; vol. 46: pp. 107 - 117.

Weizenbaum, Joseph (1976): *Computer Power and Human Reason: From Judgment to Calculation.* San Francisco: W.H. Freeman & Company

Wikipedia. URL: http://en.wikipedia.org/wiki/File:Apple_Macintosh_Desktop.png [Lesedato 06.05.2009]